



ENCYCLOPEDIA OF COMPUTER SCIENCE AND TECHNOLOGY

计算机科学技术 百科全书

(第二版)

Second Edition

主编 张效祥

清华大学出版社

ENCYCLOPEDIA OF COMPUTER SCIENCE AND TECHNOLOGY

计算机科学技术 百科全书

(第二版)

Second Edition

主编 张效祥

清华大学出版社
北京

©清华大学出版社,2005,北京。

本书之文字、图表、照片未经著作权人许可,任何人不得翻印、复制。

版权所有,翻印必究。举报电话:010-62782989 13501256678 13801310933

图书在版编目(CIP)数据

计算机科学技术百科全书/张效祥主编.—2版.—北京:清华大学出版社,2005.11
ISBN 7-302-10594-4

I. 计… II. 张… III. 计算机科学—百科全书 IV. TP3-61

中国版本图书馆CIP数据核字(2005)第016121号

出版者:清华大学出版社

<http://www.tup.com.cn>

社总机:010-62770175

地址:北京清华大学学研大厦

邮编:100084

客户服务:010-62776969

组稿编辑:张兆琪

文稿编辑:相士俊

印装者:北京华联印刷有限公司

发行者:新华书店总店北京发行所

开本:185×260 印张:84.75 插页:1 字数:2679千字

版次:2005年11月第2版 2005年11月第1次印刷

书号:ISBN 7-302-10594-4/TP·7185

印数:1~3100

定价:198.00元

第一版前言

半个世纪以来,计算机科学技术以磅礴之势迅猛发展,它以非凡的渗透力与亲合力,深入人类活动的各个领域,对人类社会的进步与发展产生了巨大的影响。计算机应用于科学研究,大大增强了人类认识自然与开发、改造和利用自然的能力,促进了现代科学技术的发展;计算机应用于生产,大大提高了人类物质生产水平和社会生产率,促进了经济的发展;计算机应用于社会服务,大大扩大和改善了服务范围与质量,提高了工作效率,推动了社会进步;计算机应用于社会文化,为人类创造文化提供了现代化工具,改变了人们创造和传播文化的方式、方法和性质,大大扩展了人类文化活动的领域,丰富了文化的内容,提高了质量;计算机进入办公室、家庭和为个人所拥有,正改变着人们的工作方式和生活方式。计算机科学技术对一个国家在政治、经济、科技、文化、军事、国防等方面发展的催化作用和强化作用,都具有难以估量的意义。它已在世界范围内形成一种现代文化。计算机科学技术在即将来临的新世纪中,必然会成为人类的重要基础文化知识之一。《计算机科学技术百科全书》(以下简称《全书》)就是迎接时代的需要,为推动我国计算机教育的普及与深入,为提高全民族计算机科学技术文化素质,为促进计算机学术研究与产业发展而撰写的。

50年来,计算机已发展为范畴宽广、内涵丰富的科学技术和规模恢宏的新兴产业。计算机与通信的融合和全球性联网,更赋予它未可限量的发展前景。《全书》力求涵盖计算机科学技术发展50年来的主要成就,广收博取,深入浅出,以通俗精练的文体,比较系统全面地介绍学科的基本知识,既适于各行各业广大读者查询,也可供计算机专业人员参阅,并作为向深广发展的桥梁与阶梯。

《全书》根据计算机学科的内在联系、相关程度与性质特点,划分为“计算机科学理论”、“计算机组织与体系结构”、“计算机软件”、“计算机硬件”、“计算机应用技术”和“人工智能”6大分支,按4级框架,共设置1293个条目200多万字。由于中文信息处理是我国及全球汉字通用地区计算机应用中的重要技术,特在“计算机应用技术”分支中,设置有关中文信息处理条目80余条,以供读者查阅。《全书》按照不同层次与内容涉及范围,将条目释文分为大、中、小3类。在释文中有一定释义的常用名词术语还择要列作“主题词”者共约1031个,与条目一起编入内容索引中,以利查阅。《全书》“总论”全面综览了计算机科学技术的内涵与对人类社会发展的巨大作用与深远意义,以引导读者全面、科学地认识计算机科学技术。《全书》设附录3种。

《全书》的撰写,着意于以简练、概括的笔触给每个条目以确切的定义和明确、完整的内容,取材于肯定成熟的知识,言必有据,确切可信。

《全书》在中国计算机学会、我国多所著名院校和计算技术研究所以及清华大学出版社等的支持和直接参与下,于1993年6月着手开展工作,至1996年12月完成全部编审,历时3年有余。有分布于全国的约400名专家、教授参与了撰写和审稿。《全书》是我国计算机学术界一部集体智慧的巨著,实现了大家多年来的共同愿望。

对《全书》不足之处,欢迎广大读者不吝赐教。

《计算机科学技术百科全书》编撰委员会

1997年2月

第二版前言

《计算机科学技术百科全书》(以下简称《全书》)第一版自 1998 年出版以来颇受各界关注。它已成为我国计算机学术、教育、产业与应用等诸领域工作者和广大计算机爱好者常用的、得力的工具书。由于计算机科技发展迅猛,自第一版发行至今 6 年多的时间里,又有了许多新内容涌现。为适应广大读者及时获得计算机科技领域更新知识的需求,从 2000 年开始筹划,在第一版的基础上进行了增、删与改写,并对《全书》框架作了适当调整,历时 4 载有余,方完成了《全书》第二版的编撰出版工作。

计算机网络在推进我国国民经济发展与社会信息化的过程中有着十分重要的作用。近年来计算机网络技术发展十分迅速,内涵丰富,为此,在《全书》第二版中将其列为独立分支。在原有的“计算机科学理论”、“计算机组织与体系结构”(第二版改名“计算机组成与体系结构”)、“计算机软件”、“计算机硬件”、“计算机应用技术”和“人工智能”6 大分支基础上增加了“计算机网络”分支。第二版在各个分支中除了适当增加新条目外,还对第一版框架中一些过于细小条目进行了合并、改写与删节,对一些已现陈旧的条目则予删除,以使整个框架更加合理,更能反映计算机科技的新近发展。第二版对“计算机科学技术总论”也做了一些修改与补充。全书共收录条目 1 381 条,比第一版略有增加。其中,“计算机科学理论”172 条、“计算机组成与体系结构”133 条、“计算机软件”320 条、“计算机硬件”162 条、“计算机网络”245 条、“计算机应用技术”244 条、“人工智能”105 条。

《全书》第二版力求对每个条目赋予简明而确切的定义以及较为明确而完整的内涵,取材于肯定成熟的知识,言必有据,确切可信。

《全书》第二版广大作者撰写条目备极艰辛,清华大学出版社各位编辑精心校勘,于此均表感谢。

对《全书》第二版不足之处,欢迎广大读者不吝赐教。

《计算机科学技术百科全书》编撰委员会

2004 年 12 月

凡 例

1. 本书是全面、系统、科学地介绍计算机科学技术知识的专业性百科全书。全书按7个学科分支(计算机科学理论、计算机组成与体系结构、计算机软件、计算机硬件、计算机网络、计算机应用技术、人工智能),4级框架的层次设置了1381个条目。它们构成了计算机科学技术完整的知识主题系统。

2. 条目分类目录由条目的题名组成,它们按学科分支的框架层次编排,体现了计算机科学技术所涵盖知识的内在联系、相关程度和性质特点。可以说,它是全部条目的系统表。

目录中有的条题名没有注明页码,表明该条题没有释文。设置这些条题的目的是为了完整地表示一个学科分支的科学体系。例如,计算机硬件分支中的“计算机逻辑部件”条题名。

目录中有的条题名会在两个分支内出现,说明它在两个学科分支体系中都有其作用。例如“自然语言处理”条题名在计算机应用技术分支和人工智能分支中均存在。

3. 本书条目按条题名汉语拼音字母顺序排列。第一字同音时,按其音调的四声顺序排列;同音同调时依次按笔画多少和笔顺排列;如完全相同,则按第二字,余类推。非汉字开头的条题名排在汉字条题名之后。依次为英文字母、希腊字母和阿拉伯数字开头的条题名,它们分别按字母顺序和数的顺序排列。

4. 全书的“参见”体系有两类情况:一类是两个条目释文内容相同但有两种条题名,则在一个条题下有释文,对另一个条题则注明“参见×××”。例如,“静止图像的压缩编码标准”条题内注明了“参见**图像的压缩编码**”。另一类是在一个条目的释文中阐述的部分内容另有专条论述或与其有关,则以“参见”方式表示。若参见的条题名在释文中出现,则该条题名用黑体字排出;若条题名在释文中不出现,则加括号,并在括号里注明参见的条题名,同时用黑体字排出。

5. 释文内用魏碑字体排出的主题词(如**微程序**)是未被本书列为条目而在文中有定性叙述或较多阐释的知识主题。

6. 本书的检索系统有:条目汉语音序索引、条目外文索引和内容索引。

内容索引中包含了全部条题名和释文内的主题词。这些主题词用魏碑字体标示,有的层次标题又是主题词,则其字体不变。

7. 书末有3个附录。附录Ⅰ为书中出现的部分英文科技名词的缩略语。附录Ⅱ为计算机及相关学科科技期刊名录,包括中国期刊和外国期刊。附录Ⅲ为计算机及相关学科学术团体名录。

8. 书中科学技术名词采用全国科学技术名词审定委员会公布的名称。尚未公布的则采用本专业中惯用的名称。

书中采用由国家技术监督局发布的《量和单位》中所规定的物理量及其单位。对一些非法定计量单位则依惯例全书统一,并给出与法定计量单位的换算关系。对计算机技术中常用的单位做到全书统一,如用 B 表示 byte, b 表示 bit, KB 表示 1 024 byte。

外国人名已有通用中文译名者,如牛顿、傅里叶,按译名写出;其余的在文中第一次出现时加括号写出原文。对专业性较强的条目,其中外国人名不予翻译。

计算机科学技术总论

基 本 概 念

计算机是一种现代化的信息处理工具。它对信息进行处理并提供所需结果,其结果(输出)取决于所接收的信息(输入)及相应的处理算法。

计算机科学技术是研究计算机的设计与制造和利用计算机进行信息获取、表示、储存、处理、控制等的理论、原则、方法和技术的学科。它包括科学与技术两方面。科学侧重研究现象与揭示规律;技术则侧重研制计算机及使用计算机进行信息处理的方法与技术手段。科学是技术的基础与依据,技术是科学的应用与体现;技术得益于科学,它又向科学提出新的课题。科学与技术相辅相成、互为作用,二者高度融合是计算机科学技术的突出特点。

计算机科学技术除了具有较强的科学性外,还具有较强的工程性,因此,它是一门科学性与工程性并重的学科。计算机科学技术的迅猛发展,除了源于微电子学等相关学科的发展外,主要源于其应用的广泛性与强烈需求,它已逐渐渗透到人类社会的各个领域,成为经济发展的倍增器,科学文化与社会进步的催化剂。应用是计算机科学技术发展的动力、源泉和归宿,而计算机科学技术又不断为应用提供日益先进的方法、设备与环境。

计算机科学技术发展迅速,还因为科学技术成果逐步转化为商品,形成开发、生产、销售、服务、培训配套的、年销售额以万亿美元计的全球性巨大的计算机产业。产业是计算机科学技术发展的依托。计算机科学技术为产业提供新思想、新方法、新技术、新工艺,更新产品,拓宽市场,增强竞争力。而产业则为科学技术的研究开发提出课题,提供资源,从而构成计算机事业发展的良性循环。总之,计算机、计算机科学技术、计算机产业三者的关系是:计算机是计算机科学技术的基本研究对象,是计算机产业的基本产品;计算机产业是计算机与计算机科学技术的依托;计算机科学技术则是计算机、计算机产业发展的生命源泉。三者又同时限定和制约于经济发展、社会发展以及相关学科的发展。

巨 大 作 用

计算机是 20 世纪 40 年代人类的伟大创造。它对人类社会的进步与发展作用巨大,影响深远。

1. 开拓了人类认识自然、改造自然的新资源

人类最早认识和开发的是物质资源,把它转化成材料,制作出简单的工具,从事个体、家庭或小作坊式的生产,这种生产方式导致生产率低下,以致社会经济发展缓慢,形成几千年的农业社会自给自足的自然经济。18世纪以蒸汽机发明为代表的产业革命兴起,开始了能量资源的开发和利用,把它转化为动力,制造出各种自动的机器作为生产工具,有效延伸了人的体能,劳动生产率显著提高,使人类进入大规模生产的工业化时代,形成以商品生产与交换为标志的市场经济。工业化对人类创造了巨大的财富,促进了社会经济的繁荣与发展,但同时也带来了非再生物质资源和能量资源的大量消耗与浪费。人类发现信息这一战略资源,还是近几十年的事。现代科学技术的进步,特别是计算机的出现,使人类从此有了自动化、信息化和一定智能化的强大工具,以开发利用信息资源,把它转化为知识产品,促使物质生产水平和社会劳动生产率空前提高,开创了信息时代的新纪元。以计算机为核心对信息资源的开发和利用,使物质资源和能量资源的效益得以更加充分、高效地发挥,人们能以合适的物质和能量创造出高质量产品,其增值来源于信息和知识。计算机与通信的融合,建立大量信息网络和大规模高速互联网,必将深刻影响人类的生产方式与生活方式,形成以信息和知识产品为特征的“信息经济”和“知识经济”。计算机的出现,使人们在物质和能量两大战略资源外,开发和利用了“信息”这一新的战略资源,开拓了人类认识自然、改造自然的新资源。

2. 增添了人类发展科学技术的新手段

长期以来,人类发展科学技术依靠两大传统手段,即理论与实验。这两种手段起过并继续起着基本作用。而计算机的出现,由于其自动、高速进行大量运算的能力和计算的精确性,致使过去科学家穷毕生精力无法办到的事,如今在短短几小时,甚至几分钟内即可变成现实,并能获得单纯依靠理论与实验难以得到的结果。从而,一方面,使传统物理学、化学、生物学等基础科学的研究进入了新的境界,出现了计算物理学、计算化学、计算生物学、计算力学等新兴学科;另一方面,在诸如电机工程、电子工程、土木工程、建筑工程、化学工程、航空工程、材料工程等工程性学科的研究中,由于利用了计算机这一现代信息处理工具以及计算机科学技术的研究成果,更新了研究手段,加速了它们的发展。同时,由于计算机科学技术与其他学科的融合,出现了人工智能、计算机图形学等交叉学科。此外,计算机用于自然资源开发、重大工程建设与环境保护等方面,正在起着越来越大的作用。计算机已在航空航天、资源勘探、大范围中长期天气预报、材料、遗传工程、核能利用、尖端武器设计等众多领域大量应用,取得了重大经济效益和社会效益。随着计算机应用的不断拓广与深入,以及计算机科学技术的不断发展,必将出现更多的新兴交叉学科。计算机和计算机科学技术的出现,在理论与实验两大传统手段外,又增添了人类发展科学技术的新手段,即计算手段。

3. 提供了人类创造文化的新工具

文化是人的行为以及体现在思想、言语、行动、制作中的成果的总体式,是人类创造的

社会精神财富和物质财富的总和。计算机用于辅助教育,丰富了教育方法。计算机辅助教育以生动的画面和动画图形来描述数学、物理、化学、历史、地理与语文等学科内容,寓教育于娱乐,以形象补充文字,提高了学习者的积极性。计算机辅助教育通过学习者与计算机之间的交互活动,使学习者能自主探索,按需学习,从而培养了学习者的创造思维和主动学习能力,收到传统教育方法难以收到的效果。以计算机为核心的电子照排系统,从文稿起草、编辑、版面编排,到制版印刷,连续完成一系列工序流程,大大提高了文化传播的能力与水平。随着电子印刷的推广,大量图文资料进入计算机硬盘、光盘等存储媒体,自然地引发了电子图书和数字图书馆的出现,几十卷的巨著存入光盘,既便于携带、查阅,又大幅度降低了出版成本。多媒体技术和超文本结构的引入,更将使电子图书、电子报章成为文化传播的手段。计算机进入美术、影视等领域已成现实。用计算机设计创作的编织、刺绣、服装、地毯、壁纸、工业造型与动画等已进入市场。在计算机上直接完成乐谱制作,用计算机设计舞蹈表演和人体动作等,已引起音乐家、舞蹈家和体操教练等的重视。机器翻译与语言文字识别等技术的进展,将在国际合作和科技文化交流等方面发挥着重大作用。在各类学校中,计算机都列为必修课程。社会上各行各业的工作人员由于学会使用计算机,其工作成果与工作效率大大提高。从而,计算机及其使用已成为人类必需的文化内容,成为与语文和数学等同等重要的基础知识。计算机的出现,为人类创造文化提供了新的现代化工具。它改变了人们创造文化的活动方式、方法和性质;拓宽了文化活动的领域;丰富了文化的内容;提高了质量;革新了传播手段;改善了学习条件;增强了传播能力,使之达到前所未有的水平。

4. 引起了人类的工作方式与生活方式的变化

计算机进入办公室、家庭和个人之手,使人类的工作方式与生活方式经历着巨大变化。社会与经济的发展使各类社会组织如政府机关、企业事业部门、金融商业机构、社会团体等的业务信息急剧增长,决策处理科学化和时效性要求大大提高,传统的工作方式与方法已不能保证质量要求和决策水平。计算机技术、通信技术与各种办公设备相结合,使人类的工作方式与方法产生了巨大变革。人们用计算机进行文字处理;用电子报表、图形、图像、声音等多媒体来表示工作中复杂、生动的实际情况;用电子邮件保证部门间信息传递的及时、方便与可靠;电子会议改进了会议方式,减少了会务工作,提高了会议效率。计算机、通信网络与各种计算机信息系统相结合,大大增强了人们掌握工作全局情况和综合分析判断的能力,有效地提高了决策、经营和管理水平。计算机进入家庭给家庭生活带来巨大变化。人们用计算机管理家庭日常事务;对家用设备如照明、煤气、空调、电源,以及门户、烟火安全等进行监控;计算机及其网络技术应用于商务活动,实现电子商务;应用于政务活动实现电子政务;应用于医疗业务,实现远程医疗;应用于教育培训,实现远程教育;计算机与电视、电话相结合,可获得电视游戏、电视点播等服务;通过计算机网络,实现在家办公。笔记本计算机与个人数字助理的发展更使计算机成为易于携带的个人手中的计算工具,并通过通信网络为实现不拘地域、空间均能开展工作提供了条件。总之,计算机、计算机网络等给人类的工作方式与生活方式带来深刻变化,并由此步入信息化社会。

发展历程

1. 国际

(1) **电子计算机的诞生** 用于计算的机器可追溯到 17 世纪,那时欧洲的一些数学家就设计制造出纯机械式的数字运算机器。著名的有 1642 年法国数学家 B. Pascal 制成的十进制加法器,1673 年德国数学家 C. N. Leibniz 研制的进行十进制数乘、除运算的计算器。在此基础上英国数学家 C. Babbage 于 1822 年研制成可以运转的差分机模型,1834 年他又设计了一种程序控制的通用分析机,但限于当时的技术条件,未能实现。在 Babbage 分析机之后的几十年间,数字式计算机的研究出现了停滞,但有一批物理学家用物理方法探求计算工具的新途径,兴起了模拟计算机的研制。模拟计算机借助连续物理量运算求解问题,用物理过程来模拟数学方程的解算过程。直到 20 世纪 30 年代模拟计算机仍受重视,但其专用性、低精度、可靠性与稳定性较差等弱点限制了它的推广。另有一类计算机曾在电子计算机出现之前起过重要作用,即 19 世纪末叶由统计工作者 H. Hollerith 等人创造的高级分类统计机。它以机电相结合的结构,采用穿孔卡片作为数据载体,完成分类、统计、制表等一系列计算操作过程。这类机器在 20 世纪 40 年代后逐渐被淘汰。最早采用电气元件研制计算机的是德国工程师 K. Zuse,他于 1941 年完成全继电器式通用计算机 Z-3。其他著名的继电器式计算机尚有由 H. Aiken 于 1944 年完成的 MARK-I 及 1947 年完成的 MARK-II。科学技术的进步,特别是电子学的迅速发展和第二次世界大战对先进计算工具的迫切需求,为现代电子计算机的诞生奠定了社会与技术基础。首先采用电子技术实现的数字计算机为 1946 年 2 月美国宾夕法尼亚大学莫尔学院制成的 ENIAC,它含有 18 000 个真空管,运算速度达到当时继电器式计算机的 1 000 倍,但它没有采用二进制操作和存储程序控制,未具备现代电子计算机的主要特征。1945 年 3 月, J. von Neumann 领导的小组发表了二进制的程序储存式的电子数字自动计算机 EDVAC 方案,1945 年 7 月, J. von Neumann 等人又提出更为完善的设计报告,宣告了现代计算机结构思想的诞生。但由于种种原因,直到 1951 年 EDVAC 才告完成。而英国剑桥大学的 M. V. Wilkes 在 EDVAC 方案的启发下于 1949 年制成的 EDSAC 成为世界上第一台程序储存式的现代电子计算机。

(2) **器件更新作为计算机划代的标志** 计算机硬件的发展受到电子开关器件的极大影响。为此,器件更新被作为计算机技术进步划代的一种标志。第一代电子管计算机(从 20 世纪 40 年代中期到 50 年代末期)。除了前述的 ENIAC 和 EDSAC 外,最具代表性的尚有 1951 年的 UNIVAC-I 和 1956 年的 IBM 704 等。1951 年由 J. P. Eckert 和 J. Mauckly 主持设计的 UNIVAC-II 是美国批量生产的第一台电子管商用计算机。为军事需要而研制的大型计算机则有 1954 年的 NORC 等。电子管计算机体积大、功耗大、故障率高,运算速度只在每秒一两万次左右。第二代为晶体管计算机(从 20 世纪 50 年代中后期到 60 年代中期),其主要特征是采用晶体管作为开关元件。和第一代的电子管计算机相比,第二代晶体管计算机具有体积小、可靠性高、功耗低、运算速度快(可达每秒执行百

万条指令)等优点。最初出现的晶体管计算机为 1956 年美国军用的 Leprechan,而美国麻省理工学院于 1957 年完成的 TX-2 对晶体管计算机的发展起了重要作用。这一代计算机的产品主要有 IBM 7040、IBM 7070、IBM 7090 等。小型计算机则有 IBM 1401。主要的大型计算机有 UNIVAC-LARC, IBM Stretch 以及 CDC 6600 等。第三代计算机(从 20 世纪 60 年代中后期到 70 年代初中期)以集成电路作为基础器件,这是微电子与计算机技术相结合的一大突破。从而可以廉价构成运算速度快、容量大、可靠性高、体积小、功耗小的各类计算机。主要有 IBM 360 系列与 PDP-11。第四代计算机(20 世纪 70 年代中后期以来)的主要特征是普遍采用大规模集成电路与超大规模集成电路技术,从而导致计算机硬件价格急剧下降,机器的性能价格比迅速提高。

(3) **计算机应用方式的发展** 在计算机出现初期,所处理的大都是科学计算和工程计算问题,计算量大而数据量相对较少,主要采用批量处理方式。20 世纪 50 年代后期,企业应用逐渐开展,数据处理问题日益增多,这类问题的数据量大,输入输出频繁,计算量相对较少,致使运算部件经常处于空闲状态。为使价格昂贵的计算机资源得以充分利用,以提高计算机系统的实际使用功效,出现了分时处理方式与交互作用方式。70 年代微处理器的出现与 80 年代微型计算机的大发展,使计算机得以进入各行各业、家庭和个人之手,大大加速了计算机的普及应用,出现了所谓个人计算方式。90 年代以来,计算机网络蓬勃发展,大量计算机联入不同规模的网中,大大扩展和加速了信息的流通,增强了社会的协调与合作能力,使计算机的应用方式向分布式和网络式发展。

(4) **计算机产品的发展** 计算机发展初期,主要针对具体应用需求研制机器,因此,型号多而产量少。有一定批量的工业生产始于 20 世纪 50 年代前期。随着应用和计算机工业的发展,人们注意到计算机产品继承性的重要性。50 年代后期出现了具有一定兼容关系的计算机系族,IBM 700、IBM 7000 系族为其代表。在这一时期还因为工业、商业、金融业等对数据处理应用的需求,促使小型计算机如 IBM 1401 及 PDP-8 等的发展。1964 年 4 月 IBM 公司发布 IBM 360 系列,对计算机的普及和大规模工业生产产生了重大影响。IBM 360 以统一的体系结构、操作系统、输入输出接口,以及科学计算、数据处理、实时控制等广阔的应用方面,达到大、中、小型计算机之间的兼容,实现了系统的通用化、系列化与标准化,成为计算机发展中的重要策略。CDC、UNIVAC、Burrough 等公司也都相继推出了系列化产品。系列机大量节约了后继机种的开发成本,缩短了开发周期。尤为重要的是,保护了用户的软件资源积累。20 世纪 70 年代初,Intel 4004 芯片研制成功,为 80 年代微型计算机的大发展奠定了基础,掀起了计算机普及的浪潮。特别是 80 年代后期 RISC 芯片的出现,使芯片运行速度大大提高,90 年代中期的 RISC 处理机每秒可执行几亿条指令。另一方面,由于众多的应用领域要求超高性能的计算机,在 1970 年前后,相继出现了 CDC 7600、STAR-100、ASC 等巨型计算机,其主要特点是:速度快、并行处理能力强。1976 年,Cray 公司推出了 Cray-1 向量巨型机,具有 12 个功能部件,运算速度达每秒 1.6 亿次浮点运算(这种巨型机又称为超级计算机,后来又统称为高性能计算机)。随着超大规模集成电路与微处理器技术的长足进步和现代科学技术对提高计算能力的强烈需求,20 世纪 80 年代以来,并行处理成为研究的热点。特别是,使用数以千计的微处理器组成的并行处理系统,其峰值运算速度已达每秒几十万亿次。

(5) **计算机软件的发展** 计算机软件的发展受到应用和硬件发展的推动和制约。反之,软件的发展也推动了应用和硬件的发展。软件的发展经历了如下阶段:从第一台计算机上的第一个程序开始到实用的高级程序设计语言出现以前为第一阶段(20世纪40年代中期到50年代中期)。如前所述,在计算机发展初期,应用领域较窄,主要是科学计算与工程计算。处理对象是数值数据。编制程序所用的工具是低级语言。程序的设计和编制工作采用个体工作方式,强调编程技巧。研究对象是顺序程序。这一阶段主要研究科学计算与工程计算程序、服务性程序和程序库。当时人们对和程序有关的文档的重要性尚认识不足,重点考虑程序本身。那时虽尚未出现“软件”一词,但毕竟由于程序是软件的主体,从发展的连续性来看,仍应将其归为第一阶段。从实用的高级程序设计语言出现以后到软件工程提出以前为第二阶段(20世纪50年代中期到60年代后期)。虽然早在1951年瑞士学者H. Rutishauser就提出设计高级语言及其翻译程序,但直到1956年在J. Backus领导下,才就IBM 704机器研制出第一个实用的高级语言FORTRAN及其翻译程序。此后,相继又有多种高级语言问世,著称者有ALGOL 60, COBOL, SIMULA, ALGOL 68等,从而设计和编制程序的功效显著提高。为了充分利用系统资源,出现了操作系统(如IBM 360操作系统)。为了适应大量数据处理问题的需要,研制了数据库及其管理系统。在20世纪50年代后期人们逐渐认识到和程序有关的文档的重要性,因此到了60年代初期,出现了“软件”一词,融程序及其有关文档为一体。这时,软件的复杂程度迅速提高,研制周期变长,正确性难以保证,可靠性问题相当突出。到了60年代中期,出现了人们难以控制的局面,即所谓软件危机。为了解决这一危机,人们进行了以下三方面的工作:第一,提出结构程序设计方法;第二,提出用工程方法开发软件;第三,从理论上探讨程序正确性和软件可靠性问题。这一阶段的研究对象增加了并发程序,并着重研究高级程序设计语言、编译程序、操作系统以及各种应用软件。计算机系统的处理能力得到提高,设计与编制程序的工作方式逐步转向合作方式。从软件工程提出迄今为第三阶段(20世纪60年代后期以来)。由于大型软件的开发是一项工程性任务,采用个体或合作方式不仅效率低、产品可靠性差,而且很难完成,只有采用工程方法才能适应。从而在1968年的大西洋公约学术会议上提出了“软件工程”的概念。三十多年来,软件领域工作的主要特点是:第一,随着应用领域的不断拓广,出现了嵌入式应用及其软件;为了适应计算机网络的需要,出现了网络软件;随着微型计算机的推广,分布式应用和分布式软件得到快速发展。第二,软件工程发展迅速,开发方式逐步由个体合作方式转向工程方式,形成了“计算机辅助软件工程”。除了开发各类工具与环境,用以支持软件的开发、运行与维护外,还有一些实验性的软件自动化系统。第三,致力研究软件体系结构、基于构件的软件、起重要支撑作用的中间件以及软件过程本身,研究各种软件开发风范与模型。第四,除了软件传统技术继续发展外,人们着重研究以智能化、自动化、集成化、并行化、开放化以及自然化为标志的软件开发新技术。第五,致力研究对象技术与主体技术。第六,注意研究软件理论,特别是软件开发过程的本质。

2. 国内

(1) **中国古代的贡献** 中国古代在计算理论与计算工具方面贡献突出。主要发明有

五：一为二进制的位。其表示符号为“爻”。爻分阴爻和阳爻。阴爻对应 0，阳爻对应 1，易经中的八卦和六十四卦分别为 3 个爻和 6 个爻的集合。德国数学家 Leibniz 曾说：“伏羲在其推演的八卦中使用了二进制算术”。二为十进制记数系统。据殷墟甲骨文和周代青铜器上的铭文记载，十万以内的自然数可由 1~9 的 9 个符号和表示十、百、千、万位值的 4 个符号表示。较当时巴比伦和古埃及的记数制更为科学。东周末年，出现了算筹体记数法，中国早就把零当作数。公元 1 世纪的《九章算术》中已阐明了负数的运算规则，印度在公元 7 世纪才提到负数，欧洲到 17 世纪才有论述负数的著作。三为筹算。筹算利用算筹作为运算工具，春秋战国时期已广泛使用，对中国古代社会的发展起了重要作用。四为珠算。它以算盘为计算工具，在元代已广泛使用，明代传至日本、朝鲜等国。五为提花机，这是一种提花织物的纺织机械，要织的图案先做成“花本”，用花本去控制织造过程。提花机是一种过程控制的纺织机械，秦汉时期已经出现，法国到 18 世纪才用穿孔卡片机控制织造过程。

(2) 中国计算机系统的研制 早在 20 世纪 50 年代初期，中国即有从事计算机研究的科研组。

中国计算机事业创始于 20 世纪 50 年代中期。1956 年国家制定《1956—1967 年科学技术发展远景规划》，将“计算技术的建立”列为紧急措施之一。一面派人去苏联考察、学习，一面在国内开办训练班，积极培养人才，同时筹建中国科学院计算技术研究所。并以苏联资料为蓝本，分别于 1958 年与 1959 年研制出我国最早期的计算机，即 103 小型数字计算机和 104 大型通用数字计算机。此后开始自主研制。1960 年由中国科学院计算技术研究所研制出小型电子管计算机 107 机，1964 年 5 月和 10 月由中国科学院计算技术研究所和华东计算技术研究所分别研制出大型电子管计算机 119 机和 J-501 机。1965 年南京大学与华东计算技术研究所合作在 J-501 机上配制了 ALGOL 语言。中国科学院计算技术研究所在 119 机上配制了 BCY 语言。1965—1966 年间中国科学院计算技术研究所、哈尔滨军事工程学院、华北计算技术研究所、华东计算技术研究所等单位分别研制出晶体管计算机：109 乙机、441B 机、108 机和 X-2 机。此外，投入生产的还有 121 机和 112 机。从而中国进入了晶体管计算机的时代。这些机器一般都配有 ALGOL 或 FORTRAN 语言。FORTRAN 语言是由长沙工学院于 1973 年首先在 441B 机上配制的。中国集成电路计算机的研究始于 1965 年。直到 1971 年，中国科学院计算技术研究所的 111 机和华北计算技术研究所的 112 机才基本研制成功。1973 年北京大学与北京有线电厂合作研制出百万次级的 150 机，华东计算技术研究所也研制出性能和 150 机相当的 655 机，并先后投入运行。这些机器都配有高级语言与管理程序。1973 年初，原第四机械工业部主持研制 100 系列与 200 系列计算机。前者与 NOVA 机兼容。清华大学负责研制的 130 机与 140 机批量生产千余台，后者指标和 IBM 360 类似，但和 IBM-360 不兼容，也生产若干台，并配有 14 个软件系统，其中包括三个操作系统（南京大学研制 XT-1，北京大学研制 XT-2，华北计算技术研究所研制 XT-3），FORTRAN（北京有线电厂主要研制），COBOL（南京大学主要研制），BASIC（西安交通大学主要研制），系统程序设计语言（南京大学研制），以及光笔等其他软件。此外，中国科学院计算技术研究所研制成 757 向量机与 KJ 8920 大型机。国防科技大学先后于 1983 年及 1992 年研制成向量式巨型机银河 I 和银河 II 以及后来的

银河Ⅲ、银河Ⅳ,它们都配有操作系统、高级语言编译程序等系统软件,这些机器对国防建设与国民经济建设均起了重要作用。另一方面,清华大学开发出中华学习机,生产十余万台。长城计算机公司与清华大学联合研制的 0520 机是我国最早的国产微型计算机。随着对微型计算机需求量的日益增加,我国计算机的装机量从 1978 年的 500 台猛增到 1990 年的 50 万台,1996 年的 500 万台,至 2003 年我国已生产微型计算机 3 000 多万台,计算机得到更为广泛的普及应用。此外,国家智能计算机研究开发中心于 1995 年研制成大规模并行计算机曙光 1000。其后又研制出曙光 2000 和曙光 3000,上海大学于 2000 年研制出大规模并行机自强。至 2003 年,联想集团的深腾 180 和 6800 以及曙光 400L 等高性能计算机系统 Linpack 值均超过每秒 1 万亿次。

(3) **中国计算机的应用** 中国计算机应用的发展可分为如下阶段。20 世纪 50 年代末至 60 年代中为第一阶段。其特点是,所解问题多为科学计算与工程计算问题,诸如求代数方程的近似解,求线性代数方程组的数值解,以及求常微分方程组、偏微分方程组的数值解等,处理对象均为数值数据。应用领域涉及国防建设、气象数值预报、工程设计等。程序人员使用低级语言编制程序,单纯手工方式,有一些简单的标准程序库与服务性程序。60 年代中至 70 年代末为第二阶段。这时所解算的问题除了科学计算与工程计算问题外,出现了数据处理问题。如前所述,这类问题计算量相对较小,数据传输量却很大,输入输出频繁,处理对象主要还是数值数据。应用领域除前述者外,还涉及各种企业、事业部门,应用面不断拓广,开发了不少管理信息系统。程序人员普遍使用高级语言,各类机器一般都配有 ALGOL, FORTRAN, COBOL, PASCAL 等语言,以及各种操作系统等系统软件,解题环境得到改善。培养了一批系统软件及应用软件开发人员。这一阶段利用计算机解题的水平显著提高。80 年代迄今为第三阶段。其主要特点是,处理对象除了数值数据外,出现了非数值数据。既有数值问题,也有逻辑问题,应用面大大拓广。所解问题涉及国防建设、国计民生、教育文化、安全保卫和娱乐健康等方面。计算机辅助技术用于辅助设计、辅助制造、辅助教育以至辅助软件工程。在软件开发过程中也尽量利用了计算机系统。软件技术与人工智能技术相结合,出现了一些富有特色的计算机辅助设计系统、专家系统以及图形、图像识别与处理系统,具有一定智能的软件工具等。40 多年来,中国计算机应用的发展已逐步从面向专业人员朝面向非专业人员过渡,由处理单纯数值对象发展为既处理数值对象又处理非数值对象,并发展了包括语言、文字、图形、图像、声音等在内的多媒体应用。计算机的应用面和应用水平正在不断拓展和提高。

(4) **中文信息处理** 中文信息处理是我国与全球汉字通行国家、地区和汉字使用者在计算机应用中面临的重大问题。

中文和西文的差异较大。40 多年来,特别自 20 世纪 70 年代中期以来,我国在中文信息处理方面进行了大量的研究开发工作。从汉字属性分析研究、汉字键盘输入技术、汉字字模技术、汉字输出技术、汉字编码以及储存、检索、软件汉化到中文篇章识别、汉语言语识别、手写汉字识别、篇章理解与处理、机器翻译、电子照排、印刷出版、中文平台等方面,取得了一系列重大成果。

对汉字编码输入,人们从汉字本身体现的各种特点出发,提出了数百种方案,实际使用者有十多种。但汉字输入问题的完善解决,尚有许多探索研究工作要做。为了储存大

量中文篇章,需解决信息压缩问题,为此提出了基于数学理论而又颇富实效的压缩技术。中文信息检索方面异彩纷呈,提出了各种中文信息检索方法。关于软件汉化,也做出了很好的工作,如汉化 DOS、汉化 Unix 等。

中文篇章的识别、理解与处理比较困难。为此,必须进行词切分,从语法与语义两方面联系起来考察。多年来,在汉语语法方面开展了大量研究工作。在研制具体汉语处理系统的同时,还从理论上探讨了汉语语法的形式化问题。用合适的形式体系来描述受限汉语的语法,取得了较大进展。同时对难度更大的汉语语义的形式化问题也进行了探索。此外,在与中文篇章的理解与处理密切相关的语料库方面,也进行了卓有成效的工作。

汉语言语识别的难度很大,既要考虑汉语的平、上、去、入四声,又要考虑到上、下文,目前有的系统已初步具有学习功能,经过对发音者的语音学习后,即可识别出同一发音者的言语,而且达到较高的正确率。对手写汉字的识别也很困难。首先要确定字体(如楷、宋、隶、篆、草体以及简、繁体等);其次要考虑到各人写法的可允许差别范围。必须从识别方案、特征抽取、识别算法等多方面探求解决方法。目前已出现一些实验性系统,在特定使用领域并加以若干限制的条件下,有的系统已臻实用。此外,如机器翻译、电子照排、印刷出版等,我国均有出色的成果,并有相当影响的产品问世。近年来,还在将中文信息处理系统的硬件支撑与软件支撑联系统一考虑,建立计算机系统的中文平台方面做出了不少成绩。

基 本 内 容

计算机科学技术的基本内容可概括为计算机科学理论、计算机组成与体系结构、计算机软件、计算机硬件、计算机网络、计算机应用技术以及人工智能等领域。

1. 计算机科学理论

计算机科学理论包括数值计算、离散数学、计算理论和程序理论四部分。数值计算讨论用于模拟物理过程或社会过程的各种算法的设计、分析和使用。早在 18 世纪与 19 世纪,高斯、牛顿、傅里叶等著名数学家就研究过数值计算方法,而计算机的诞生更大大促进了数值计算的发展。数值计算涉及的内容颇多,如方程求根、数值逼近、数值微分、数值积分、数值代数、线性代数方程组的数值解法、矩阵特征值计算、微分方程数值解法等。例如,高次代数方程求根的常用方法有二分法、牛顿法、割线法等。数值微分讨论求导数近似值的理论与方法,常用的有有限差分法。数值积分讨论求定积分近似值的理论与方法。梯形法和辛普森法均为世人所熟知。线性代数方程组的数值解法用以求线性代数方程组的数值解,通常有直接法和迭代法两类。高斯消去法为直接法,简单迭代法和赛德尔迭代法均为迭代法。离散数学是泛指数学中讨论离散对象的分支。和连续数学不同,离散数学通常涉及整数系,由于数字计算机是离散机,离散数学的重要性不言而喻。通常认为离散数学包括集合论、图论、组合学、数理逻辑、抽象代数、线性代数、差分方程、离散概率论等学科。图论是研究图的性质的学科。而图论中的图并非初等数学中的图,后者只是连续函数的图形,图论中的图却是一组顶点(结点)和一组连接两顶点的边(支)所构成的集合。组合论讨论计算某类对象个数的方法,它在统计学、理论物理、化学、社会科学、通信

理论以及计算机科学技术中均有重要作用。多数组合论问题可归结为存在性问题、枚举性问题或选择性问题。数理逻辑研究形式体系。作为其组成部分的命题演算与谓词演算等在计算机科学技术中作用巨大,影响深远。诸如计算机设计、软件开发、程序正确性验证,以及人工智能等领域无不用到数理逻辑。抽象代数讨论离散对象结构,它在计算机科学技术中应用广泛。例如,半群已用于形式语言理论和自动机理论,群在编码理论中有其重要作用。线性代数虽然涉及实变量,但其结构与处理均为离散,因而,也可归为离散数学。此外,差分方程,离散概率论等亦属离散数学。计算理论主要包括算法、算法学、计算复杂性理论、可计算性理论、自动机理论、形式语言理论等。算法是解题过程的精确描述,它包括有限多个规则,并具有如下性质:第一,将算法作用于特定的输入集或问题描述,可导致由有限多个动作构成的动作序列;第二,该动作序列具有惟一一个初始动作;第三,序列中的每一动作具有一个或多个后继动作(序列中的末一动作的后继动作可视为空动作);第四,序列或者终止于问题的解,或者终止于某一陈述,以表明问题对该输入集而言不可解。算法学是系统研究算法的学科。通常包括设计、验证以及分析三部分。设计是创建算法的过程,并研究良好的创建方法;验证在于证明算法的正确性,基本途径是数学归纳法;分析着重确定算法的效用,当一问题有多种算法可用时,则比较其相对效用。计算复杂性确定从数学上提出的问题的固有难度,通过研究计算复杂性,可以断定哪些问题是固有困难的,从而有助于寻求更为优越的算法。算法复杂性是针对特定算法而言,最佳算法复杂性等于计算复杂性。计算复杂性理论则是用数学方法研究各类问题的计算复杂性的学科。它在计算机科学技术中既有理论意义,又有实用价值。可计算性理论是研究计算的一般性质的数学理论。它通过建立计算的数学模型,精确区分哪些问题是可计算的,哪些问题是不可计算的。计算的过程就是执行算法的过程。可计算性理论主要包括图灵机、丘奇图灵论题、 λ 演算、原始递归函数、部分递归函数、递归集、递归可枚举集、可判定性等。自动机理论是研究称作自动机的抽象理想机的数学学科。自动机是信息处理设备(如计算机)的抽象。多数自动机都是图灵机的特例。自动机理论一般包括有限自动机理论、无限自动机理论、概率自动机理论、细胞自动机理论等。形式语言理论是用数学方法研究自然语言(如英语)和人工语言(如程序设计语言)的语法的理论。形式语言就是模拟这些语言的数学工具。它只研究语言的组成规则,不研究语言的含义。内容包括描述工具、文法分类(如乔姆斯基层次)、语言分类,以及各类语言的性质及其间的关系等。程序理论研究程序的语义性质、语用性质和程序的开发,主要包括程序语义理论、程序语用理论、数据类型理论、程序逻辑理论、程序验证理论、并发程序设计理论和混合程序设计理论等。程序理论和计算理论是计算机科学理论的两大支柱。形式语义理论是用数学方法研究程序语言语义的理论,包括操作语义、公理语义、指称语义以及代数语义等。此外,还有旨在用计算机研究代数演算的“计算机代数”以及用计算机研究数学证明的“计算机数学”等。

2. 计算机组成与体系结构

计算机体系结构着重研究计算机系统的物理或硬件结构、各组成部分的属性以及这些部分的相互联系。它可分为系统体系结构和实现体系结构两个方面。前者着重从系统

软件开发人员的角度看计算机系统的功能行为和概念结构;后者从计算机系统的性能特征和价格出发,考虑该系统的结构和实现,包括中央处理器、存储器等部件的结构和实现。也有人认为计算机体系结构专指系统体系结构,而将实现体系结构称为计算机组成。本书的计算机组成与体系结构包括上述的计算机系统体系结构和计算机实现体系结构。其内容还包括计算机类型、计算机 RAS 技术和计算机性能评价等。可从不同角度来区分计算机类型。按计算机内数据表示的方式分,有数字计算机、模拟计算机、混合计算机等。按系统规模和性能分,有微型计算机、小型计算机和高性能计算机(一般是多机系统或并行处理系统)等。按用途分,有通用计算机和专用计算机,通用计算机能够处理各种不同类型的问题,专用计算机只适合于处理某一类特定问题。按工作风格分,有基于冯·诺依曼结构的传统计算机和不基于冯·诺依曼结构的非传统计算机。传统计算机的特征是控制驱动,即指令执行的次序由指令计数器控制,存储器按地址访问;非传统计算机可以是数据驱动或需求驱动,或不按地址访问存储器。计算机组成包括数据表示、算术逻辑运算、指令系统、中央处理器、存储器组织和输入输出技术。数据表示包括二进制制、浮点数标准和字符集;算术逻辑运算包括二进制算术运算和逻辑运算;指令系统包括指令类型、指令格式和寻址方式;中央处理器包括运算器、控制器、数据通路等;存储器组织包括各种存储器、存储器的差错校验以及对存储器的性能评价;输入输出技术是主机和输入输出设备连接的技术,包括总线、输入输出通道、输入输出接口等。计算机体系结构包括处理机体系结构、存储系统、并行处理系统、分布式处理系统、高性能计算系统、网格计算、开放系统以及系统兼容性等。处理机体系结构包括各种类型的处理机结构,特别是精简指令集计算机的体系结构对计算机的发展有重要的影响。存储系统具有层次结构。一般由四级存储器组成:第一级是寄存器(在中央处理器中);第二级是高速缓冲存储器;第三级是主存储器;第四级是辅助存储器。这四级存储器都是实际存储器。虚拟存储器为用户提供比主存储器容量大得多的可随机访问的地址空间。并行处理系统旨在突破单机运算速度与作业吞吐量的限制,以适应日益增长的巨大计算能力需求。它将多个处理机或多个功能部件通过内部网络连接起来,实现并行处理。其体系结构大体上可分为单指令流多数据流和多指令流多数据流两种。已实现了多种形式的并行处理系统。分布式处理系统将不同地点或不同功能的多台计算机用通信网络连接起来,协同完成信息处理任务。它包括客户-服务器计算、计算机簇、分布式异构型计算机系统等。高性能计算系统用高速计算机系统求解复杂问题,它包括向量计算、大规模并行处理、集群式计算等。计算机 RAS 技术包括计算机系统的可靠性、计算机的易用性和易维护性、计算机安全等。计算机性能评价包括对运算速度的评价、评价系统性能的指标、性能评价的基准程序以及对计算机系统性能的模拟等。

3. 计算机软件

计算机软件一般指计算机系统程序及其文档,也可以指在研究、开发、维护以及使用上述含义下的软件所涉及的理论、方法、技术所构成的学科。软件的作用有三:一是用作计算机用户与硬件之间的接口界面;二是在计算机系统中起指挥管理作用;三是作为计算机体系结构设计的重要依据。软件的发展过程大致可分为三个阶段。从第一台计算

机上第一个程序的出现到实用的高级程序设计语言出现以前(20世纪40年代中期至50年代中期)为第一阶段。从实用的高级程序设计语言出现以后到软件工程出现以前(20世纪50年代中期至60年代后期)为第二阶段。软件工程出现以后迄今(20世纪60年代后期以来)为第三阶段。一般说来,软件可分为系统软件、支撑软件以及应用软件三类。系统软件是计算机系统中最靠近硬件层次的软件,如操作系统、编译程序等均为系统软件。它和具体的应用领域无关,解任何领域的问题一般都要用到系统软件。支撑软件是支撑其他软件的开发、运行与维护的软件,例如:软件开发环境与中间件等均为支撑软件。应用软件是特定应用领域的专用软件,如人口普查软件、飞机订票软件等。上述分类并非绝对,而是相互有所覆盖交叉和变动,三者既有分工,又相结合,不能截然分开。软件的基本内容包括软件语言、软件方法学、软件工程以及软件系统。软件语言是用以书写软件的语言。它包括书写软件需求定义的需求级语言、书写软件功能规约的功能级语言、书写软件设计规约的设计级语言、书写实现算法的实现级语言以及书写软件文档的文档语言。软件方法学是以软件方法为研究对象的学科。从开发风范上看,有自顶向下的软件开发方法以及自底向上的软件开发方法。从表现形式上看,有形式方法与非形式方法。从适用范围来看,有整体性方法与局部性方法。软件工程是应用计算机科学与数学原理制作软件的工程。它含有四个要素:第一为目标,如产品的正确性、易用性以及价格合宜等。第二为风范,它反映软件开发过程的原则与风格。风范是模型的基础,模型是风范的体现。一般有功能分解风范、功能综合风范等。第三为过程,它主要包括需求、设计、实现、确认以及支撑等阶段。第四为原则,它主要涉及系统设计、软件设计、软件过程支撑以及软件过程管理等方面。如认识需求的变动性,采用稳妥的设计方法,提供高水平的支撑,提供有效的管理等。软件系统包括操作系统、中间件系统、数据库系统、语言处理系统、分布式软件系统、网络软件系统及人-机交互软件系统等。操作系统用以管理系统资源,旨在提高计算机的总体效用。一般包括存储管理、设备管理、信息管理、作业管理等。中间件系统居于操作系统、数据库系统和应用程序之间的层次,用以支撑应用程序的运行。数据库系统包括数据库及其管理系统。数据库是相互关联的在某种特定的数据模式指导下组织而成的各种类型的数据的集合。数据库管理系统则是为数据库的建立、使用和维护而配置的软件,它建立在操作系统的基础上,对数据库进行统一的控制和维护。它一般包括模式翻译、应用程序的编译、查询命令的解释执行以及运行管理等部分。语言处理系统包括各种类型的语言处理程序,如解释程序、汇编程序、编译程序、编辑程序、装配程序等。分布式软件系统用以管理、支撑分布式计算系统。它一般包括分布式操作系统、分布式程序设计语言及其编译程序、分布式数据库管理系统、分布式算法及其软件包、分布式开发工具包等。网络软件系统是在计算机网络环境中,用于支持数据通信和各种网络活动。它主要包括通信软件、网络协议软件和网络应用系统、网络服务管理系统以及用于特殊网络站点的软件等。人-机交互软件系统是人-机交互系统中的软件子系统,它一般包括人-机接口软件、命令语言及其处理系统、用户接口管理系统、多媒体软件、超文本软件等。

4. 计算机硬件

计算机硬件是构成计算机系统的所有物质元器件、部件、设备以及相应的工作原理与

设计、制造、检测等技术的总称。集成电路是微电子学和制造工艺技术高度发展的产物,它是将大量晶体管、二极管、电阻、电容等各种元件集成在一块半导体芯片上,以实现特定完整功能的器件。元器件包括集成电路、印制电路板及其他磁性元件、电子元件等。部件和设备包括控制器、运算器、存储器、输入输出设备、电源等。控制器用以控制整个计算机自动地执行程序指令。它由指令部件、时序部件和操作控制部件三部分组成。控制器调动计算机各个部件参与运行,依次接受程序指令,进行解释,并把相应的控制信号送往各个部件以完成规定的操作。运算器用以实现二进制编码的算术与逻辑运算,由算术逻辑部件、累加器和通用寄存器等组成。运算器在控制器的控制和存储器的支持下,完成程序的算术和逻辑运算。存储器用来储存程序所需的信息。根据不同的功能、结构与工作原理,存储器可分为半导体存储器、磁盘存储器、磁带存储器、光盘存储器等。过去曾一度使用的磁心存储器和磁鼓存储器等则随着技术的发展而被淘汰。输入输出设备是计算机和用户的交互接口部件,主要包括输入设备、输出设备以及终端设备等三大类。输入设备有批式输入设备(如纸带输入机、软盘输入机等),交互式输入设备(如键盘、鼠标器、触屏等)以及言语、文字、图形输入设备等。输出设备有显示设备、印刷设备、言语输出设备、绘图仪等。终端是用户与网络进行交互操作以利用其计算机资源的一种设备,通常可分为两类:一类是通用终端,适用于一般用户。另一类是面向特定作业的终端,如商业收款机、银行柜员机、信用卡验证终端等。

此外,计算机硬件还可包括计算机制造、计算机检测及计算机维护等技术。

5. 计算机网络

计算机网络是地理上分散的多台自主计算机互连的集合。自主性排除了网络系统中的主从关系,互连需遵循约定的通信协议。计算机网络可实现信息交互、资源共享、协同工作以及在线处理等功能。

自 1969 年美国国防部的国防高级研究计划局 DARPA 建立了全世界第一个分组交换网 ARPANET(即 Internet 的前身)以来,Internet 的发展促进了信息技术的发展,信息市场的开拓,并大大加速了社会信息化的进程。计算机网络有多种分类法,如按地域范围分,有局域网和广域网;按拓扑结构分,有总线网、星状网和环状网等。网络的主要技术有数据通信、体系结构与协议、互连、管理以及网络安全等。其发展趋势的总目标是要在各个国家,进而在全球建立完善的信息基础设施,以使一个国家的信息网络能使任何人在任何地点、任何时间,将任何形式的信息传递给任何地点的任何人。计算机、通信、信息内容三种技术的融合以及电信网、电视网、计算机网的合一是支持建立完善的信息基础设施的重要途径。

近十年来,以 Internet 为代表的计算机网络技术迅速发展,其应用日益广泛、深入,对人们的生活、学习、工作和人际交往方式的改变产生了极为深远的影响。从传统的电子邮件到网上的实际对话和网络会议,为人们提供了极为方便、快速的交互通信工具。网络提供程序、数据和信息资源的共享,从根本上改变了用户使用计算机资源的环境和方式。遍布在 Internet 上的全球性的超媒体系统——万维网成为人们快速、有效、方便地获取信息的重要工具。网络在各个领域的应用广度和深度正在发展,协同工作可以进行没有时空

限制的协同研究,大大加速了科学的发明和发展;远程教学正在改变传统的教学模式和校园文化;电子商务改变了传统的商业模式和商贸关系;电子政务是实现管理型政府向服务型政府演变的一种创新的政务管理模式。随着网络应用不断向纵深发展,终将出现网络应用无“孔”不入、处处可见的局面。

随着计算机网络及其应用的不断发展,网络的开放性、信息的共享性、应用的广泛性等导致世界范围内网络攻击手段层出不穷,网络犯罪日趋严重,如不积极研究出有效保障网络安全的方法与手段,必将给人类带来莫大灾难。因此,网络安全问题极应予以高度重视。网络安全是在分布网络环境中,对信息载体(处理载体、存储载体、传输载体)和信息处理、传输、储存、访问提供安全保护,以防止数据、信息内容、能力被非授权使用和篡改或拒绝服务。本质上,安全就是风险管理。风险是外部威胁和内部漏洞(脆弱性)的综合结果。漏洞有可能存在于计算机系统和网络中,也可能存在于管理过程中。

6. 计算机应用技术

计算机应用技术着重研究计算机用于各个领域所涉及的原理、方法与技术。范围相当广泛,其中内容丰富并已发展为系统领域者有中文信息处理、计算机图形学、数字图像处理、计算机辅助技术、多媒体计算技术、计算机控制、计算机信息系统以及计算机仿真等。中文信息处理研究用计算机处理中文信息所涉及的原理、方法和技术。由于中文和西文(汉语和西语)差别很大(如西文为拼音文字,二十几个字母,而汉语常用字就有六七千个,总数达五万余;西文词与词间有空格,而中文字字相连;西文多有形态,如性别、时态、数量的变化,而中文甚少;汉语文法亦不如西语规范等),致使中文信息处理不能完全沿用西文信息的处理方法。为了推广应用计算机,加速实现社会信息化,中文信息处理技术对我们炎黄子孙以及进行国际文化交流均有特殊重要意义。中文信息处理技术的主要内容有:在国际标准架构下,建立、发展全球通用的汉字编码字符集、汉字编码输入、汉字识别、汉语言语识别、机器翻译、自然语言理解、中文信息检索、电子印前处理等。计算机图形学是借助计算机产生真实物体图形或虚构物体图形的理论、方法和技术。计算机辅助造型和画面绘制是计算机图形学的两个重要组成部分。计算机图形学的关键性技术主要有造型技术、人机交互技术、彩色生成和处理、真实感图形生成技术、动画以及科学计算形象化技术等方面。计算机图形学具有代表性的应用领域有:计算形象化、动画设计、计算机辅助设计与制造、过程控制、办公自动化、电子出版以及艺术制作等。数字图像处理是利用计算机将模糊或受损图像进行处理以实现图像增强、复原、重建以及分割、配色等的过程与技术。主要包括图像的获取与输入、图像储存、图像处理、图像表现、图像分析与识别、图像输出与传输等方面。数字图像处理已广泛应用于卫星遥感遥测、大地测量、资源勘探以及医学等方面。计算机辅助技术的应用十分广泛,主要有计算机辅助设计、计算机辅助制造、计算机辅助工程、计算机辅助教学等。计算机辅助设计是利用计算机帮助设计人员进行工程、产品等设计工作的过程和技术。由计算机辅助产生的设计结果通过图形设备与设计人员进行交互,以便及时对设计做出判断和修改,最终完成设计工作。计算机辅助设计有效地减轻了设计人员的劳动,缩短了设计周期,提高了设计质量。计算机辅助制造是在制造业中利用计算机通过各种设备辅助完成产品的加工、装配、检测和包装等

的过程和技术。它已广泛用于飞机、汽车、机械、家用电器、电子产品等制造业,显著提高了企业的生产效率和产品质量,缩短了生产周期,降低了产品成本。计算机辅助工程是利用计算机帮助工程人员进行工程分析与实现的过程和技术。它可以提高产品质量,缩短工程周期以及降低产品成本等。其应用领域有机械工程、土木工程、电子工程等。计算机辅助教学是利用计算机辅助教师对学生进行治疗、训练的过程和技术。一般通过学生与计算机的对话实现。对话在计算机指导程序和学生之间进行。学生可以根据个人特点进行学习,变被动学习为主动学习,教学形象直观,从而提高学习效果。多媒体计算技术指用计算机综合处理文字、图形、图像、声音等多种形式的媒体信息,使多种信息建立起逻辑连接,并集成为系统的技术。它汇集计算机体系结构、计算机软件以及视频音频信号获取、处理和显示输出等技术。它的出现拓宽了计算机处理信息的类型,方便了用户,推动了计算机的普及应用,促进了计算机科学技术与其他学科的发展。它一般包括:多媒体硬件支撑、多媒体软件、音频视频信号的压缩和编码、多媒体数据库以及多媒体通信等。计算机控制是计算机用于实验、生产或类似的过程中进行操作控制的过程和技术。它通过检测获取受控对象的数据和变量信息,经计算做出判断,实现控制。计算机控制能有效提高产品质量和数量,降低能耗和材耗,改善劳动条件和提高操作安全性。计算机信息系统在这里特指主要任务是对数据进行采集、储存和处理并以人机交互方式为用户提供信息服务的系统。通常它处理的数据量大,且绝大部分是持久的、可以为多种应用所共享。它一般具有数据管理功能,并能向用户提供信息检索、统计、事务处理、规划、决策等信息服务。应用最普遍的是事务处理系统和信息储存与检索系统。目前流行的计算机信息系统的体系结构主要有客户-服务器结构、多级服务器的客户-服务器结构和对等(对称)服务结构等三种。当前,计算机信息系统正朝向系统集成化、结构分布化、功能智能化、服务公用化的方向发展。计算机仿真是对各种类型的系统,根据它们的有关概念、变量、规则、逻辑关系、数学表达式、图形和表格等必要信息,建立数学模型或描述模型并在计算机上加以体现和试验,从而达到分析、研究该系统之目的的过程和技术。计算机仿真主要有离散事件系统仿真、一体化仿真、连续系统仿真以及仿真语言等。计算机仿真已成为工程设计,系统开发,自然科学、经济和社会问题研究以及进行教育训练等的有力手段。

7. 人工智能

人工智能着重研究、解释和模拟人类智能、智能行为及其规律。其主要任务是建立智能信息处理理论,进而设计并实现可以展现某些近似于人类智能行为的计算系统。对于智能与智能行为虽然迄今尚无一致的理解,但一般认为智能主要指人的学习能力。在研究路线上,初期有微观结构路线与智能行为路线之分。至20世纪50年代末,后一路线成为研究的主流。人工智能的研究内容一般可分为基础问题、系统问题和应用问题。基础问题包括认知基础与技术基础。前者涉及常识知识、学习、联想及问题求解等;后者涉及表示、推理及搜索等。系统问题涉及知识库、推理机及分布式系统结构等。应用问题涉及自然语言处理、软件自动化、智能机器人以及各类专家系统等。知识表示研究用以表示智能系统中所用的知识的语言以及用以实现相应语言的方法和技术。常用的知识表示方法有两类:一类是过程性表示,将问题写成一组过程;另一类是陈述性表示,它又可分为谓

词表示与结构化表示。此外,近年来又出现了联结机制表示、基因表示等。推理是利用显式储存的知识以产生另外显式知识的过程。推理过程亦可分为两类,即演绎推理与非演绎推理。归纳推理、连接机制推理、类比推理、不确定性推理等均为非演绎推理。问题求解研究寻找或构造问题的解,其显著特点是,如何将待解问题对应于一个解空间,设法对解空间进行搜索,以找出待解问题的满意解。因此,问题求解和搜索密切相关。搜索是系统在引导推理时所使用的策略,以便有效地找出问题的解。搜索有两类:一类称为盲目搜索(又称无信息搜索);另一类称为启发式搜索(又称有信息搜索)。后者与待解问题的领域知识密切相关。常识知识指的是凭直觉(即生活经验)来认识世界的那些非专业性知识。常识不一定是普遍真理,但它有时对求解某些问题却可能起决定作用,关键在于如何正确表示常识,如何恰当使用常识,这些问题的难度均很大。机器学习是指在特定表示的前提下借助计算机系统进行学习,以产生系统中原先不存在的知识或提高系统解题能力或执行某种事务的能力。机器学习按其机制不同可分为:归纳学习、类比学习、分析学习、联结机制学习以及遗传学习等。由于学习能力是智能的核心,机器学习就不言而喻成为人工智能的核心问题之一。但迄今真正用于实际的具有学习能力的计算机系统尚为数不多。自然语言处理包括自然语言理解和自然语言生成。自然语言理解是自然语言处理的基础,如果计算机系统对自然语言不理解,当然也就谈不上能处理自然语言。任何语言均有语法、语义以及语用三个方面。语法反映语言的结构,不涉及其含义;语义表示语言 and 情景无关的含义;语用表示语言 and 情景有关的含义。自然语言的完整理解应包括对语法、语义以及语用三方面的理解。目前研究较多的是语法理解,语义理解次之,语用理解的研究工作尚开展得不多。自然语言生成是自然语言理解的逆过程,犹如计算机图形学是计算机图像处理过程的逆过程一样。智能机器人是一个跨学科领域,它涉及规划、推理、学习等方面。机器人的研制在国民经济建设与国防建设中均有巨大作用。总之,人工智能已显示出它的旺盛生命力,它在计算机科学技术及社会发展中所起的作用将日益显著。

计 算 机 产 业

计算机产业不属本书重点阐述范围,但计算机科学技术与计算机产业关系密切。产业是以产品开发、生产、销售、服务为主旨的各种公司与机构所形成的总体。计算机产业则是以计算机系统为产品的产业。它包括计算机制造业、计算机服务业以及计算机软件产业。计算机制造业从事计算机系统的生产制造。属于计算机制造业的企业有各种系统制造厂,外围设备和终端设备制造厂,记录媒体制造厂以及提供专用的应用系统的厂家等。计算机服务业是为满足使用计算机或信息处理的需要而提供服务的行业,它一般包括处理服务、专业服务和系统集成等方面,也包括计算机和有关设备的租赁、修理和维护等。计算机软件产业是以计算机软件的开发、生产、销售与服务为主旨的产业,它兼有制造业与服务业的双重特性。各种类型的计算机软件及相应机构都属于计算机软件产业。计算机产业可为国民经济和社会各方面带来巨大的经济效益和社会效益,其水平和规模已成为衡量国家经济实力和军事实力的重要标志。

世界第一台计算机于1946年研制完成,但到20世纪50年代前期才在美国开始有少数原来从事办公或商用机器制造的公司(如Remington Raml、IBM等)开发生产出最早期

的计算机商品(如 UNIVAC, IBM 650 等)。50 年代后期至 60 年代前期,美、英、日等国的计算机公司相继创立(如 CDC、DEC、Ferranti、富士通、东芝等),计算机产业开始从美国扩展到欧洲和日本,产品主要为面向军用或科学计算的大型机,产量较小,未形成大规模计算机产业。60 年代中期,以 IBM 360 通用机系列及 DEC 的 PDP 小型机系列为代表的大批量生产使计算机产业进入大规模生产的发展阶段。70 年代初期,以 Intel 为代表的微处理器芯片问世,并随之于 70 年代中期开发出微型计算机系统,进入市场,由此大大拓展了计算机的应用,促进了计算机产业的空前发展。同时,也使集成电路的研制生产企业成为计算机产业的重要组成部分。在计算机产业发展的相当长时期内,计算机系统的硬件和软件一直是以封闭方式由同一企业配套开发的。因此,二者密不可分。80 年代初期,IBM 采用 Intel 芯片与 Microsoft DOS 操作系统,开发出 PC 微型计算机系统,引起众多厂家进入所谓兼容机的角逐。软件开发已不再与硬件开发紧密结合。80 年代,由于微型计算机、工作站和局域网技术的发展,更有力地推动了计算机产业的发展,使其地位迅速上升,成为各国经济发展中重要的支柱产业。进入 90 年代,世界计算机产业进入了一个新的发展时期。软件产业异军突起,并以其技术发展牵动硬件技术的发展。到 1995 年,世界计算机产量已超过 5 000 万台,销售总额 5 000 亿美元以上。其中软、硬件产品销售额基本相当。同时,由于精简指令集计算机等新技术的出现,世界计算机产业结构、产业布局和技术方向也发生了重大变化。技术发展一改过去专有封闭的做法,而走开放标准之路;产业也由过去由某一公司软件、硬件垂直开发的结构转变为软、硬件逐渐分离,并且多采用国际分工、专业化开发生产的结构。在 20 世纪的最后 10 年中,计算机产业由于计算机网络大发展导致:第一,微型计算机的需求量猛增;第二,多种领域(如电视、电影、电话、微机 etc)联合创建新产品,其特点是,体积更小、功能更强、使用更方便,如个人无线通信系统、手提计算机、智能信用卡等;第三,计算机产业在各国的国民经济中的比重猛增。至 20 世纪末计算机产业已成为一种具有战略意义的产业。

中国计算机产业的发展可分为三个阶段。第一阶段(20 世纪 50 年代中期到 70 年代末期)的重点是根据国防建设与科学研究的需要,从借鉴苏制样机研究仿制,逐步走向独立自主开发。科研成果即产品,为专门应用部门使用。因此,当时计算机生产厂家少、规模小、产品少而分散、发展缓慢。第二阶段(20 世纪 80 年代初期到 90 年代初期)国内进口了国外的微型计算机,在此基础上开发出 0300 系列、0500 系列等国产微型计算机。遵循引进、消化、开发、创新的方针,开发出一些小型计算机与工作站,以及 CC-DOS 汉字操作系统,发明了多种汉字输入与处理方法,汉化了 IBM、DEC 等公司生产的机器上使用的 VMS 和 DOS/VSE、MVS 等操作系统,开发了大量的应用软件和应用软件包。国内市场规模迅速扩大,计算机的年销售额由 1981 年的 5.2 亿元人民币增到 1990 年的 55.5 亿元,计算机产业有了较大发展。第三阶段(20 世纪 90 年代以来)的显著变化是国际各大公司纷纷进入中国市场。国内企业向两极发展,一是扩大规模,继续发展自己的产品;一是向应用方向发展,针对用户需求开发应用系统。并自主开发 Office、Linux 等软件,进入市场。同时,中外合资企业及外资独资企业的大量出现,使外向型产业的规模迅速扩大,致使国内市场规模更快扩大,1996 年计算机产业的市场销售额为 920 亿元,2000 年为 2150 亿元,2003 年为 3327 亿元。中国计算机产业的总体水平迅速提高,规模迅速扩大,软件产业及信息服务业迅速发展,产业结构渐趋合理,基本形成了制造业、服务业和软件产业。

它们在计算机产业中的比重日益增加。

发展展望

1. 计算机科学技术与通信科学技术紧密融合,相互渗透,大大加速人类社会信息化进程

随着世界各国信息基础设施的建立与发展,计算机科学技术与通信科学技术更加紧密融合,相互渗透,全球性的计算机联网促进信息资源的开发和利用。计算机进入千家万户,使它成为人类工作与生活的必需品。计算机科学技术成为人类必须学习的基础知识。特别是,网络计算、移动式计算、嵌入式计算、多媒体技术、虚拟现实技术、面向对象技术等有机结合与综合应用,展示出计算机与计算机科学技术的宏伟前景,必将出现计算机、计算机网络、计算机应用几乎处处可见的局面,从而大大加速人类社会信息化进程。

2. 新型元器件和体系结构的发展,以及相关技术的发展,大大提高计算机系统的性能,便于人类更好地认识自然

随着微电子加工技术的发展,半导体集成电路将会有更高的速度、更高的集成度和更高的性能价格比。计算机体系结构的发展将使计算机获得更高的性能。过去一些无法解决的复杂问题将有可能解决,使人类能更好地探索自然和驾驭自然。

3. 新技术的研究、开发与利用,大大提高计算机软件的功能与性能,解决计算机系统开发中的软件瓶颈问题

随着以智能化、集成化、自动化、并行化、开放化以及自然化为标志的计算机软件新技术的深入研究、开发与利用,不仅使软件的功能与性能迅速提高,而且有可能从根本上解决软件生产率低下的问题。结合软件工程实践,探讨软件理论,有可能从理论上弄清软件开发的复杂度,进而采取有效措施进行控制,从理论与实践两方面来解决计算机系统开发中的软件瓶颈问题。

4. 信息安全保密等成为计算机与计算机科学技术领域的重大课题

在全球联网的趋势下,为保证信息资源共享,计算机系统与网络的互操作性、开放性和标准化将受到高度重视。同时由于计算机进入千家万户,成为人人可以利用的设施,使用的简明化、自然化和信息安全保密等将成为计算机与计算机科学技术领域中的重大课题。

电子数字计算机是20世纪40年代人类的伟大创造。半个多世纪以来,计算机、计算机科学技术、计算机产业在世界范围内蓬勃发展,规模空前。它的诞生和发展对人类社会作用巨大,影响深远。今后宜继续本着造福于全人类的宗旨,遵循促进社会发展的方向,按世界各国相互学习、取长补短、互利互惠、共同提高的原则,阔步前进,为人类做出更大的贡献。

总论编写组
(徐家福执笔)

2004年10月

条 目 分 类 目 录

I . 计 算 机 科 学 理 论

数值计算 (numerical computation)	715
数值计算误差分析 (error analysis of numerical computation)	717
数值逼近 (numerical approximation)	712
插值 (interpolation)	48
数值微分 (numerical differentiation)	719
数值积分 (numerical integration)	713
切比雪夫逼近 (Chebyshev approximation)	569
平方逼近 (approximation in quadratic norm)	558
快速傅里叶变换 (fast Fourier transform)	483
矩阵特征值问题数值解法 (numerical solution of matrix eigenvalue problems)	447
最小二乘法 (method of least squares)	1089
线性代数方程组数值解法 (numerical solution for system of linear algebraic equations)	894
矩阵计算 (matrix computation)	445
高次代数方程解法 (solution of polynomial equation)	204
非线性代数方程组数值解法 (numerical solution for system of nonlinear algebraic equations)	174
最优化方法 (optimization method)	1091
常微分方程数值解法 (numerical solution of ordinary differential equation)	55
偏微分方程数值解法 (numerical solution of partial differential equation)	553
有限元方法 (finite element method)	979
样条函数 (spline function)	942
计算几何 (computational geometry)	409
数理统计 (mathematical statistics)	710
伪随机数 (pseudo-random numbers)	865
蒙特卡罗法 (Monte Carlo method)	522
回归分析 (regression analysis)	285

排队论 (queueing theory)	548
参数估计 (parameter estimation)	40
假设检验 (hypothesis testing)	416
离散数学 (discrete mathematics)	498
集合论 (set theory)	328
集合 (set)	328
集合运算 (operations of sets)	330
映射 (mapping)	968
关系 (relation)	233
序数 (ordinal number)	929
基数 (cardinal number)	306
逻辑学 (logics)	
数理逻辑 (mathematical logic)	708
命题逻辑 (propositional logic)	529
一阶逻辑 (first order logic)	945
高阶逻辑 (higher-order logic)	205
哥德尔完全性定理 (Gödel's completeness theorem)	214
模型论 (model theory)	542
霍恩逻辑 (Horn logic)	293
多值逻辑 (multiple valued logic)	154
模糊逻辑 (fuzzy logic)	530
模态逻辑 (modal logic)	539
时态逻辑 (temporal logic)	660
线性逻辑 (linear logic)	897
组合逻辑 (combinatory logic)	1084
非单调逻辑 (non-monotonic logic)	167
直觉主义逻辑 (intuitionistic logic)	1028
代数学 (algebra)	
抽象代数 (abstract algebra)	74
群 (group)	577
环 (ring)	285
域 (field)	992
格 (lattice)	214
完全偏序 (complete partial order, CPO)	826
布尔代数 (Boolean algebra)	37
多类代数 (many-sorted algebra)	143
关系代数 (relational algebra)	234
Σ (基调)代数 (Σ (signature) algebra)	1152

计算机代数 (computer algebra)	343
计算机数学 (computer mathematics)	376
范畴论 (category theory)	159
图论 (graph theory)	786
有向图 (directed graph)	983
无向图 (undirected graph)	882
树 (tree)	679
平面图 (planar graph)	559
最短路径问题 (shortest path problem)	1089
中国邮路问题 (Chinese postman problem)	1047
计算数论 (computational number theory)	412
素数 (prime number)	758
筛法 (sieve method)	642
素性测试 (primality test)	759
最大公因子 (great common divisor)	1088
因子分解 (factoring)	959
同余 (congruence)	782
孙子定理 (Sun's theorem)	768
组合学 (combinatorics)	1085
密码学 (cryptology)	523
计算理论 (theory of computation)	410
算法 (algorithm)	760
并行算法 (parallel algorithm)	27
概率算法 (probabilistic algorithm)	199
组合算法 (combinatorial algorithm)	1084
排序算法 (sorting algorithm)	548
图论算法 (graph algorithm)	787
最小生成树 (minimum spanning tree)	1090
最大流 (maximum flow)	1088
VLSI 算法 (VLSI algorithm)	1141
算法学 (algorithmics)	763
算法设计 (design of algorithm)	762
计算复杂性理论 (computational complexity theory)	336
复杂性度量 (complexity measure)	196
时间复杂性 (time complexity)	659
空间复杂性 (space complexity)	480
复杂性归约 (complexity reduction)	197
图灵归约 (Turing reduction)	784

多项式时间归约 (polynomial time reduction)	152
多项式空间归约 (polynomial space reduction)	151
多项式谱系 (polynomial hierarchy)	151
NP 完全性理论 (theory of NP completeness)	1123
P 类问题 (class P of problems)	1132
NP 类问题 (class NP of problems)	1121
NP 完全问题 (NP complete problem)	1122
NP 完全问题近似方法 (approximate method for NP complete problem)	1123
可计算性理论 (computability theory)	470
可计算函数 (computable function)	469
原始递归函数 (primitive recursive function)	997
哥德尔配数 (Gödel numbering)	213
递归函数 (recursive function)	114
阿克曼函数 (Ackermann's function)	1
可判定问题 (decidable problem)	475
不可判定问题 (undecidable problem)	37
停机问题 (halting problem)	775
波斯特对应问题 (Post's correspondence problem)	32
自动机理论 (automaton theory)	1071
有限自动机 (finite automaton, FA)	981
下推自动机 (pushdown automaton)	888
线性有界自动机 (linear bounded automaton)	898
图灵机 (Turing machine)	784
波斯特机 (Post machine)	32
随机存取机 (random access machine, RAM)	767
栈自动机 (stack automaton)	1009
ω -有限自动机 (ω -finite state automaton)	1151
概率自动机 (probabilistic automaton)	199
细胞自动机 (cellular automaton)	887
形式语言理论 (formal language theory)	916
乔姆斯基层次 (Chomsky hierarchy)	568
文法 (grammar)	868
正则文法 (regular grammar)	1013
上下文无关文法 (context free grammar)	643
上下文有关文法 (context sensitive grammar)	644
短语结构文法 (phrase structure grammar)	137
巴克斯范式 (Backus normal form, BNF)	3

正则表达式 (regular expression)	1012
线性文法 (linear grammar)	898
乔姆斯基范式 (Chomsky normal form)	569
格雷贝奇范式 (Greibach normal form)	215
LR(k) 文法 (LR(k) Grammar)	1120
属性文法 (attribute grammar)	678
佩特里网论 (Petri net theory)	549
程序理论 (theory of programs)	65
形式语义 (formal semantics)	919
操作语义 (operational semantics)	45
指称语义 (denotational semantics)	1036
公理语义 (axiomatic semantics)	222
代数语义 (algebraic semantics)	108
论域理论 (domain theory)	513
λ 演算 (λ calculus)	1148
类型论 (type theory)	492
马丁洛夫类型理论 (Martin-Löf's type theory)	520
多态类型 (polymorphic type)	150
并发模型 (models of concurrency)	19
进程代数 (process algebra)	433
通信系统演算 (calculus of communication systems, CCS)	778
通信顺序进程 (communicating sequential processes, CSP)	777
π 演算 (π-calculus)	1150
程序逻辑 (program logic)	67
混合计算模型 (hybrid computational models)	291
混合自动机 (hybrid automaton)	292
时段演算 (duration calculus)	656
程序验证 (verification of programs)	70
模型检验 (model checking)	541

II. 计算机组成与体系结构

电子计算机 (electronic computer)	121
数字计算机 (digital computer)	736
模拟计算机 (analog computer)	535
混合计算机 (hybrid computer)	290
数字微分分析机 (digital differential analyzer)	745
计算机类型 (computer category)	365

微型计算机 (microcomputer)	860
微处理器 (microprocessor)	857
单片计算机 (single-chip computer)	110
数字信号处理器 (digital signal processor, DSP)	748
移动式计算机 (mobile computer)	947
笔记本计算机 (notebook computer)	10
个人数字助理 (personal digital assistant, PDA)	216
可穿戴计算 (wearable computing)	469
工作站 (workstation)	220
小型计算机 (minicomputer)	905
超级小型计算机 (super-minicomputer)	58
大型计算机 (large-scale computer, mainframe)	104
巨型计算机 (supercomputer)	449
嵌入式计算机 (embedded computer)	567
服务器 (server)	191
过程控制计算机 (process-control computer)	257
容错计算机 (fault-tolerant computing)	594
抗恶劣环境计算机 (severe environment computer)	465
绿色计算机 (green computer)	513
非传统计算机 (non-traditional computer)	165
数据流计算机 (dataflow computer)	698
归约机 (reduction machine)	255
逻辑推理机 (logic inference machine)	517
神经计算机 (neural computing)	651
光计算机 (optical computer)	240
生物计算 (biocomputing)	653
量子计算 (quantum computing)	507
计算机组成 (computer organization)	406
计算机运算基础 (computer arithmetic basic)	
数制 (number system)	720
浮点数标准 (floating-point number standard)	192
字符集 (character set)	1066
算术逻辑运算 (arithmetic/logic operation)	
二进制算术运算 (binary arithmetic operation)	155
逻辑运算 (logic operation)	517
指令系统 (instruction set)	1041
指令类型 (instruction type)	1039

指令格式 (instruction format)	1037
寻址方式 (addressing mode)	931
中央处理器 (central processing unit, CPU)	1051
运算器 (arithmetic unit)	1005
硬连线控制器 (hard-wired control unit)	976
微程序控制器 (micro-programmed control unit, MCU)	856
数据通路 (data path)	701
机器周期 (machine cycle)	305
时序系统 (timing system)	661
中断 (interrupt)	1047
存储器组成 (memory organization)	96
存储器类型 (memory type)	95
主存储器 (main memory, MM)	1057
辅助存储器 (auxiliary memory)	194
存储器差错校验 (memory error checking and correction)	94
存储器性能 (memory performance)	96
输入输出技术 (input/output technique)	674
系统总线 (system bus)	886
总线标准 (bus standard)	1081
输入输出通道 (input/output channel)	677
输入输出接口 (input/output interface)	674
数据传送 (data transfer)	686
直接存储器存取 (direct memory access, DMA)	1028
假脱机 (simultaneous peripheral operations online, spool)	418
模数转换器 (analog-to-digital converter, ADC)	539
数模转换器 (digital-to-analog converter, DAC)	711
终端 (terminal)	1055
计算机系统结构 (computer system architecture)	
处理机体系结构 (processor architecture)	78
复杂指令集计算机 (complex instruction set computer, CISC)	198
精简指令集计算机 (reduced instruction set computer, RISC)	433
计算机流水线 (computer pipeline)	367
多发射结构 (multiple issue architecture)	141
指令级并行处理 (instruction level parallel processing, ILPP)	1038
协处理器 (coprocessor)	905
VLIW 处理机 (very long instruction word processor)	1140
软件流水 (software pipelining)	625
网络处理器 (network processor)	833

媒体处理器(media processor)	520
存储系统(memory system)	98
存储管理(memory management)	92
存储保护(memory protection)	91
高速缓冲存储器(cache)	206
高速缓冲存储器一致性(cache coherence)	208
虚拟存储器(virtual memory)	919
转换检测缓冲器(translation lookaside buffer, TLB)	1064
联想存储器(associative memory)	506
并行处理系统(parallel processing system)	22
阵列处理机(array processor)	1012
硬件同步机制(hardware synchronization mechanism)	974
超结点结构(supernode architecture)	59
互联网(interconnection network, ICN)	281
多处理机系统总线(multiprocessing system bus)	139
共享存储(shared memory)	228
分布式共享存储(distributed shared memory)	180
消息传递(message passing)	902
路由选择(routing)	512
高性能计算(high performance computing)	212
向量计算(vector computing)	900
大规模并行处理(massively parallel processing, MPP)	101
集群式计算(cluster computing)	331
网格计算(grid computing)	827
分布式处理系统(distributed processing system)	178
客户-服务器计算(client /server computing)	478
计算机簇(computer cluster)	342
分布式异构型计算机系统(distributed heterogeneous computer system)	183
分布式计算环境(distributed computing environment)	180
开放系统(open system)	458
系统兼容性(system compatibility)	883
计算机 RAS 技术 (computer reliability availability and serviceability)	339
计算机系统可靠性(reliability of computer system)	387
容错计算(fault-tolerant computing)	594
自检验电路(self-checking circuit)	1073
平均故障间隔时间(mean time between failures, MTBF)	559
计算机病毒(computer virus)	342
计算机可用性(computer availability)	362

计算机可维护性 (computer maintainability)	361
计算机故障诊断 (fault diagnosis of computers)	355
系统维护 (system maintenance)	884
计算机安全 (computer security)	341
计算机性能评价 (computer performance evaluation)	391
运算速度评价 (arithmetic speed evaluation)	1005
数据处理速率 (processing data rate, PDR)	682
综合理论性能 (composite theoretical performance, CTP)	1079
系统性能指标 (system performance specification)	885
基准程序 (benchmark)	314
LINPACK 基准程序 (LINPACK benchmark)	1118
SPEC 基准程序 (SPEC benchmark)	1135
TPC 基准程序 (TPC benchmark program)	1137
计算机系统性能模拟 (performance simulation of computer systems)	388

Ⅲ. 计算机软件

计算机软件 (computer software)	369
软件语言 (software language)	631
需求定义语言 (requirements definition language)	926
软件需求定义 (software requirements definition)	630
PSL 语言 (PSL language)	1132
PSA 程序 (PSA program)	1131
数据流图 (data flow diagram)	699
统一建模语言 (unified modeling language)	783
功能语言 (functional language)	225
广谱语言 (wide spectrum language)	250
功能规约 (functional specification)	225
形式功能规约 (formal functional specification)	914
功能规约语言 (functional specification language)	225
CIP-L 语言 (CIP-L language)	1101
Z 语言 (Z language)	1146
FGSPEC 语言 (FGSPEC language)	1106
前后断言方法 (pre-and post-assertion method)	566
最弱前置条件方法 (weakest precondition method)	1089
设计性语言 (design language)	647
设计规约 (design specification)	646
形式设计规约 (formal design specification)	916

PDL 语言 (PDL language)	1129
状态转移图 (state transition diagram)	1064
模块结构图 (modular structure diagram)	534
实体联系图 (entity relationship diagram)	663
接口定义语言 (interface definition language)	427
SETL 语言 (SETL language)	1133
Larch 语言 (Larch language)	1117
GSPEC 语言 (GSPEC language)	1109
OBJ 语言 (OBJ language)	1125
统一建模语言 (unified modeling language)	783
程序设计语言 (programming language)	69
低级语言 (low level language)	113
机器语言 (machine language)	305
汇编语言 (assembly language)	287
高级语言 (high level language)	205
命令式语言 (imperative language)	528
系统程序设计语言 (systems programming language)	883
交互式语言 (interactive language)	424
面向对象语言 (object oriented language)	526
可扩充语言 (extensible language)	474
并发程序设计语言 (concurrent programming language)	18
并行程序设计语言 (parallel programming language)	22
FORTRAN 语言 (FORTRAN language)	1107
ALGOL 60 语言 (ALGOL 60 language)	1097
COBOL 语言 (COBOL language)	1101
BCY 语言 (BCY language)	1100
PASCAL 语言 (PASCAL language)	1129
C 语言 (C language)	1102
Ada 语言 (Ada language)	1096
Modula-2 语言 (Modula-2 language)	1121
XCY 语言 (XCY language)	1146
XYZ/E 语言族 (XYZ/E language family)	1146
BASIC 语言 (BASIC language)	1099
ALGOL 68 语言 (ALGOL 68 language)	1097
Smalltalk 语言 (Smalltalk language)	1133
C++ 语言 (C++ language)	1100
Eiffel 语言 (Eiffel language)	1105
Java 语言 (Java language)	1116

说明性语言 (declarative language)	757
函数式程序设计语言 (functional programming language)	262
逻辑程序设计语言 (logic programming language)	516
面向问题语言 (problem-oriented language)	527
数据流语言 (data flow language)	700
表处理语言 (list processing language)	15
串处理语言 (string processing language)	82
LISP 语言 (LISP language)	1119
FP 语言 (FP language)	1107
ML 语言 (ML language)	1121
Miranda 语言 (Miranda language)	1120
PROLOG 语言 (PROLOG language)	1131
PARLOG 语言 (PARLOG language)	1128
VAL 语言 (VAL language)	1140
SNOBOL 语言 (SNOBOL language)	1134
过程语言 (procedural language)	261
非过程语言 (nonprocedural language)	171
可视编程语言 (visual programming language)	475
置标语言 (markup language)	1046
标准通用置标语言 (standard generalized markup language, SGML)	15
超文本置标语言 (hyper text markup language, HTML)	60
可扩展置标语言 (extensible markup language, XML)	474
语法 (syntax)	983
语义 (semantics)	991
语用 (pragmatics)	992
元语言 (metalanguage)	996
巴克斯-诺尔形式体系 (Backus-Naur formalism, BNF)	3
语法图 (syntax diagram)	984
乔姆斯基层次 (Chomsky hierarchy)	568
0 型文法 (type 0 grammar)	1154
1 型文法 (type 1 grammar)	1154
2 型文法 (type 2 grammar)	1154
3 型文法 (type 3 grammar)	1154
产生式 (production)	52
波兰记法 (Polish notation)	31
Occam 语言 (Occam language)	1125
文档语言 (documentation language)	868
程序 (program)	63

例程 (routine)	499
子例程 (subroutine)	1066
协同例程 (coroutine)	906
子程序 (subprogram)	1065
数据 (data)	679
值 (value)	1032
常量 (constant)	54
变量 (variable)	14
表达式 (expression)	16
语句 (statement)	985
说明 (declaration)	756
函数与过程 (function and procedure)	263
数据类型 (data type)	696
简单类型 (primitive type)	419
构造类型 (composite type)	229
数组类型 (array type)	751
记录类型 (record type)	414
指针类型 (pointer type)	1043
抽象数据类型 (abstract data type)	75
类型定义 (type definition)	492
数据结构 (data structures)	688
绑定 (binding)	9
对象 (object)	138
类 (class)	489
软件构件 (software component)	615
软件主体 (software agent)	634
继承 (inheritance)	414
信息 (Information)	909
信息隐蔽 (information hiding)	912
封装 (encapsulation)	191
会合 (rendezvous)	287
异常处理 (exception handling)	958
软件方法学 (software methodology)	606
自顶向下方法 (top-down method)	1068
结构化方法 (structured method)	429
面向数据结构方法 (data structure-oriented method)	526
Jackson 系统开发方法 (Jackson system development method)	1115
数据结构化系统开发方法 (data structured system development	

method)	689
结构化分析与设计技术 (structured analysis and design technique, SADT)	430
自底向上方法 (bottom up method)	1068
面向对象方法 (object-oriented method)	524
模块化方法 (modular method)	533
原型速成方法 (rapid prototyping method)	997
基于构件的软件开发方法 (component-based software development method, CBSD)	309
形式方法 (formal method)	912
形式规约 (formal specification)	914
维也纳开发方法 (Vienna development method, VDM)	864
代数规约 (algebraic specification)	107
基于类型理论的方法 (type theory based method)	310
软件自动化方法 (software automation method)	635
程序转换方法 (program transformation method)	72
演绎综合方法 (deductive synthesis method)	941
归纳综合方法 (inductive synthesis method)	255
过程实现方法 (procedural implementation method)	259
程序设计方法学 (programming methodology)	68
程序 (program)	63
程序设计 (programming)	68
结构化程序设计 (structured programming)	429
过程式程序设计 (procedural programming)	260
逻辑程序设计 (logic programming)	515
函数式程序设计 (functional programming)	262
面向对象程序设计 (object-oriented programming)	524
顺序程序设计 (sequential programming)	754
并发程序设计 (concurrent programming)	17
并行编译程序 (parallelizing compiler)	20
分布式程序设计 (distributed programming)	177
可视程序设计 (visual programming)	476
文化程序设计 (literate programming)	869
软件工程 (software engineering)	609
软件生存周期 (software life cycle)	627
软件开发模型 (software development model)	622
瀑布模型 (waterfall model)	560
演化模型 (evolutionary model)	940
螺旋模型 (spiral model)	518

喷泉模型 (fountain model)	552
软件过程 (software process)	617
管理过程 (management process)	236
获取过程 (acquisition process)	292
供应过程 (supply process)	227
开发过程 (development process)	456
运作过程 (operation process)	1006
维护过程 (maintenance process)	864
支持过程 (supporting process)	1016
剪裁过程 (tailoring process)	418
软件过程模型 (software process model)	618
能力成熟度模型 (capability maturity model, CMM)	545
需求工程 (requirements engineering)	929
软件体系结构 (software architecture)	628
软件设计模式 (software design pattern)	627
软件开发方法 (software development method)	619
结构化方法 (structured method)	429
模块化方法 (modular method)	533
Jackson 系统开发方法 (Jackson system development method)	1115
面向对象方法 (object-oriented method)	524
原型速成方法 (rapid prototyping method)	997
基于构件的软件开发方法 (component-based software development method, CBSD)	309
软件复用 (software reuse)	608
领域工程 (domain engineering)	508
软件调试 (software debugging)	606
软件测试 (software testing)	605
软件维护 (software maintenance)	628
软件逆向工程 (software reverse engineering)	626
程序分析 (program analysis)	64
软件理解 (software understanding; software comprehension)	624
软件再工程 (software reengineering)	632
软件配置管理 (software configuration management)	626
软件工具 (software tool)	614
项目管理工具 (project management tool)	900
配置管理工具 (configuration management tool)	552
需求分析工具 (requirements analysis tool)	928
设计工具 (designing tool)	646

编码工具 (coding tool)	12
调试工具 (debugging tool)	772
测试工具 (testing tool)	46
理解工具 (understanding tool)	498
界面工具 (interface tool)	432
维护工具 (maintenance tool)	863
软件工程环境 (software engineering environment)	612
软件开发环境 (software development environment)	621
软件库 (software library)	624
工具箱 (toolkit)	218
软件包 (software package)	605
软件构件库 (software component library)	616
计算机辅助软件工程 (computer aided software engineering, CASE)	352
软件质量 (software quality)	632
软件安全性 (software safety)	603
软件可靠性 (software reliability)	623
软件工程经济学 (software engineering economics)	613
计算机软件的法律保护 (legal protection of computer software)	372
软件系统 (software systems)	629
操作系统 (operating system)	42
进程 (process)	432
线程 (thread)	893
多道程序设计 (multiprogramming)	141
任务调度程序 (task scheduler)	593
批处理 (batch processing)	553
分时处理 (time-sharing processing)	186
死锁 (deadlock)	758
文件 (file)	872
作业 (job)	1093
处理器管理程序 (processor manager)	80
存储管理程序 (memory manager)	93
文件管理程序 (file manager)	875
输入输出管理程序 (input/output manager)	673
作业管理程序 (job manager)	1093
微内核 (microkernel)	860
嵌入式操作系统 (embedded operating systems)	567
UNIX 操作系统 (UNIX operating system)	1138
DOS 操作系统 (DOS operating system)	1103

Windows 操作系统(Windows operating system)	1143
Linux 操作系统(Linux operating system)	1118
软件中间件(software middleware)	633
语言处理系统(language processing system)	989
虚拟机(virtual machine)	921
解释程序(interpreter)	431
汇编程序(assembler)	286
编译程序(compiler)	12
宏处理程序(macroprocessor)	280
编辑程序(editor)	12
装入程序(loader)	1064
连接编辑程序(linkage editor)	500
自编译程序(self-compiler)	1067
交叉编译程序(cross compiler)	423
编译程序的编译程序(compiler-compiler)	13
并行编译程序(parallelizing compiler)	20
词法分析(lexical analysis)	84
语法分析(syntax analysis, parsing)	983
代码生成(code generation)	106
代码优化(code optimization)	107
数据库系统(database system)	694
数据库管理系统(database management system, DBMS)	692
模式(schema)	536
数据共享(data sharing)	688
数据模型(data model)	700
查询处理(query processing)	49
查询优化(query optimization)	50
索引(index)	768
数据完整性(data integrity)	706
数据安全性(data security)	679
事务处理(transaction processing)	663
事务元(transaction)	664
并发控制(concurrency control)	18
故障恢复(fault recovery)	233
事务进程监控器(transaction process monitor)	663
半结构化数据(semistructured data)	9
数据库(database)	690
层次数据库(hierarchical database)	47

网状数据库 (network database)	855
关系数据库 (relational database)	235
演绎数据库 (deductive database)	941
面向对象数据库 (object oriented database)	525
对象-关系数据库 (object-relation database)	138
并行数据库 (parallel database)	27
工程数据库 (engineering database)	217
多媒体数据库 (multimedia database)	146
分布式数据库 (distributed database)	182
空间数据库 (spatial database)	481
移动数据库 (mobile database)	948
万维网数据管理 (web data management)	827
数据仓库 (data warehouse)	681
联机分析处理 (online analytical processing, OLAP)	503
数据挖掘 (data mining)	706
数字图书馆 (digital library)	743
数据库设计 (database design)	693
数据库系统三级结构 (three-level architecture of database system)	695
键码 (key)	422
规范化 (normalization)	256
范式 (normal form)	161
数据依赖 (data dependency)	707
实体联系模型 (entity-relationship model)	662
数据库性能评价 (database performance evaluation)	696
数据库连通性标准 (database connectivity standard)	693
分布式软件系统 (distributed software system)	181
分布式操作系统 (distributed operating system)	177
分布式数据库系统 (distributed database system)	182
并发控制 (concurrency control)	18
分片 (fragmentation)	186
网络软件 (network software)	842
人机交互系统 (human-computer interaction system)	589
用户界面 (user interface)	976
命令语言 (command language)	528
窗口系统 (window system)	83
用户界面管理系统 (user interface management system, UIMS)	977
多模态人机交互 (multimodal human-computer interaction)	148

IV. 计算机硬件

计算机硬件 (computer hardware)	396
集成电路 (integrated circuit, IC)	315
数字集成电路 (digital integrated circuit)	732
专用逻辑集成电路 (application specific logic integrated circuit, ASLIC)	1061
微处理器 (microprocessor)	857
微控制器 (micro-controller)	858
数字信号处理器 (digital signal processor, DSP)	748
言语 (语音) 合成器 (speech synthesizer)	934
总线控制器 (bus controller)	1082
总线仲裁器 (bus arbiter)	1082
存储器管理部件 (memory management unit, MMU)	94
半导体存储器芯片 (semiconductor memory chip)	6
随机存取存储器芯片 (random access memory chip)	766
动态随机存取存储器芯片 (dynamic random access memory chip)	134
静态随机存取存储器芯片 (static random access memory chip)	435
视频随机存取存储器芯片 (video random access memory chip)	667
只读存储器芯片 (read only memory chip)	1032
可编程只读存储器芯片 (programmable read only memory chip)	467
可擦编程只读存储器芯片 (erasable programmable read only memory chip)	468
电可擦编程只读存储器芯片 (electrically erasable programmable read only memory chip)	117
快可擦编程只读存储器芯片 (flash erasable programmable read only memory chip)	482
电源集成电路 (intergrated circuits in power supply)	120
片上系统 (system on a chip, SOC)	553
砷化镓集成电路 (GaAs integrated circuit, GaAs IC)	649
锗硅异质结器件 (heterojunction devices of SiGe/Si)	1009
光电集成电路 (optoelectronic integrated circuit, OEIC)	240
超导集成电路 (superconducting integrated circuit)	57
神经网络芯片 (neural network chip)	652
计算机逻辑部件 (computer logic unit)	
中央处理器 (central processing unit, CPU)	1051
算术逻辑部件 (arithmetic and logic unit, ALU)	764
加法器 (adder)	415

乘法器 (multiplier)	62
除法器 (divider)	76
通用寄存器 (general purpose register)	780
指令寄存器 (instruction register)	1039
程序计数器 (program counter)	64
硬连线控制器 (hard-wired control unit)	976
微程序控制器 (micro-programmed control unit, MCU)	856
计算机存储设备 (computer storage device)	
半导体存储器 (semiconductor memory)	4
半导体读写存储器 (semiconductor read-write memory)	7
只读存储器 (read only memory, ROM)	1032
固态硬盘 (solid state disk, SSD)	230
电荷耦合器件存储器 (charge-coupled device memory)	117
砷化镓存储器 (GaAs memory)	648
磁存储器 (magnetic storage)	84
数字磁记录 (digital magnetic recording)	722
数字磁记录编码方法 (encoding methods of digital magnetic recording)	725
数字磁记录检错码 (error detection codes of digital magnetic recording)	728
数字磁记录纠错码 (error correcting codes of digital magnetic recording)	729
磁盘存储器 (magnetic disk storage)	88
硬磁盘存储器 (hard disk storage)	968
硬磁盘驱动器 (hard disk drive, HDD)	971
硬磁盘控制器 (hard disk controller)	970
硬磁盘适配器 (hard disk adapter)	973
软磁盘存储器 (floppy disk storage)	600
软磁盘驱动器 (floppy disk drive, FDD)	601
软磁盘控制器 (floppy disk controller)	600
软磁盘适配器 (floppy disk adapter)	603
磁带存储器 (magnetic tape storage)	85
开盘式磁带驱动器 (reel-to-reel tape drive)	463
盒带驱动器 (cartridge tape drive)	278
流式磁带驱动器 (streaming tape drive)	511
光存储器 (optical storage)	239
光记录 (optical recording)	241
光记录编码方法 (encoding method of optical recording)	241

只读光盘驱动器 (read only optical disc drive)	1035
数字可写光盘 (compact disc-recordable, CD-R)	739
数字可重写光盘 (compact disc-rewritable, CD-RW)	740
视频光盘机 (video compact disc player)	665
数字多用途光盘 (digital versatile disc, DVD)	730
相变光盘驱动器 (phase change disc drive)	899
磁光盘驱动器 (magneto-optical disc drive)	86
光盘控制器 (optical disc controller)	244
全息照相存储器 (holographic memory)	575
外存储子系统 (external storage subsystems)	824
外存储设备接口 (external storage device interface)	822
磁盘阵列 (magnetic disk array)	89
附网存储 (network attached storage, NAS)	194
存储区域网 (storage area network, SAN)	96
自动盒带库 (automated cartridge tape library)	1070
光盘库 (optical disc library)	245
光盘塔 (optical disc tower)	245
光盘镜像服务器 (CD/DVD mirror server)	243
计算机输入输出设备 (computer input/output device)	
输入设备 (input device)	672
键盘 (keyboard)	423
鼠标器 (mouse)	677
跟踪球 (track ball)	217
数字化仪 (digitizer)	731
光笔 (light pen)	239
触屏 (touch screen)	80
控制杆 (joystick)	481
光学字符阅读机 (optical character reader, OCR)	249
光学标记阅读机 (optical mark reader, OMR)	248
条码阅读器 (bar code reader)	771
磁卡机 (magnetic card reader)	87
智能卡阅读器 (smart card reader)	1045
扫描仪 (scanner)	642
数字相机 (digital camera)	747
数字摄像头 (digital PC camera)	741
输出设备 (output device)	672
显示器 (display device)	889
击打式打印机 (impact printer)	295

非击打式印刷机(nonimpact printer)	171
绘图机(plotter)	288
终端设备(terminal device)	1055
脱机设备(off-line equipment)	820
数据准备设备(data preparation device)	707
脱机打印设备(off-line printer)	820
输入输出设备接口(input/output device interface)	674
计算机电源(power supply for computer)	344
直流电源(direct current power supply)	1029
不间断电源(uninterruptible power system, UPS)	35
大型计算机电源系统(power supply system for large-scale computer)	105
计算机工程设计和制造(engineering design and manufacturing of computers)	
计算机工程设计(computer engineering design)	
可靠性设计(reliability design)	473
热设计(thermal management, thermal design)	580
印制板设计(printed circuit board design)	963
高速数字信号传输(high speed digital signal transmission)	210
硬件验证(hardware verification)	976
计算机制造(computer manufacturing)	
集成电路制造(integrated circuit manufacturing)	323
印制板测试(printed circuit board testing)	962
绕接(wire-wrap connection)	580
无焊压接(solderless crimp connection)	878
波峰焊(wave-soldering)	30
高密度组装(high density packaging)	
微组装技术(micro packaging technology)	862
多芯片模块(multichip module)	153
表面安装技术(surface mounting technology, SMT)	16
计算机硬件测试(computer hardware testing)	
元器件测试(component testing)	994
集成电路测试程序(integrated circuit test program)	319
集成电路测试系统(integrated circuit test system)	321
印制板测试(printed circuit board testing)	962
印制板在线测试(in-circuit testing of printed circuit board)	965
电源测试(power supply testing)	119
设备测试(device testing)	645
打印机测试(printer testing)	101
软磁盘测试(floppy disk testing)	599

软磁盘驱动器测试 (floppy disk drive testing)	602
硬磁盘驱动器测试 (hard disk drive testing)	972
标准带 (standard tape)	14
调整盘 (customer engineer diskette)	772
清洗盘 (cleaning disk)	570
计算机整机检测 (computer hardware system testing)	402
计算机硬件可靠性 (computer hardware reliability)	400
元器件可靠性 (component reliability)	995
元器件老化 (component burn-in)	995
元器件筛选 (component screening)	996
加固技术 (ruggedization technology)	416
防信息泄漏技术 (technique of electro-mechanical protection against encission and spurious transmission, TEMPEST)	162
电磁兼容性 (electromagnetic compatibility, EMC)	115
计算机维护 (computer maintenance)	385
计算机机房设施 (computer room facility)	359

V. 计算机网络

计算机网络 (computer network)	380
数据通信 (data communications)	701
数据传输 (data transmission)	682
串行传输 (serial transmission)	83
异步传输 (asynchronous transmission)	955
同步传输 (synchronous transmission)	780
传输损耗 (transmission impairments)	81
传输介质 (transmission medium)	
双绞线电缆 (twisted pair cable)	751
同轴电缆 (coaxial cable)	782
光缆 (fiber optical cable)	242
无线传输 (wireless transmission)	
微波通信 (microwave communication)	856
卫星通信 (satellite communication)	866
移动通信 (mobile communication)	949
数据调制与编码 (data modulation and encoding)	
数据调制 (data modulation)	687
数据编码 (data encoding)	679
多路复用 (multiplexing)	144

时分多路复用(time division multiplexing, TDM)	657
频分多路复用(frequency division multiplexing, FDM)	557
波分多路复用(wavelength division multiplexing, WDM)	30
数据通信接口(data communication interface)	703
数据电路设备(data circuit equipment, DCE)	686
数据终端设备(data terminal equipment, DTE)	707
数据通信接口标准(data communication interface standard)	704
数据链路控制(data link control)	696
流控(flow control)	510
数据传输差错检测与控制(error detection and control of data transmission)	684
数据链路控制规程(data link control protocol)	697
交换技术(switching technologies)	426
电路交换(Circuit switching)	118
存储转发交换(store and forward switching)	99
分组交换(packet switching)	188
帧中继交换(frame relay switching)	1015
信元交换(cell switching)	912
宽带接入网(broadband access network)	487
宽带接入技术(wideband access technologies)	486
数字用户专用线(digital subscriber line, DSL)	750
电缆接入技术(cable access technologies)	118
卫星接入技术(satellite technologies)	866
宽带接入体系结构(broadband access architecture)	486
集成数字环载波(integrated digital loop carrier, IDLC)	327
混合光纤同轴电缆(hybrid fiber coaxial cable, HFC)	290
光纤到路边(fiber - to - the - curb, FTTC)	246
网络通信设备(network communication unit)	
通信控制设备(communication control unit)	776
前端处理器(front-end processor)	566
通信控制器(communication controller)	776
中继器(repeater)	1048
网桥(bridge)	854
交换机(switch)	425
路由器(router)	511
网关(gateway)	830
网络适配器(network adapter)	845
终端服务器(terminal server)	1055

通信服务器 (communication server)	776
计时器 (timer)	336
调制解调器 (modem)	774
集中器 (concentrator)	334
分组装拆器 (packet assembler/disassembler, PAD)	190
集线器 (hub)	334
收发器 (transceiver)	670
网络体系结构 (network architecture)	847
网络协议 (network protocol)	848
网络协议规范 (network protocol specification)	850
网络服务 (network services)	834
网络数据单元 (network data unit)	845
开放系统互连基准 (参考) 模型 (open system interconnection reference model)	461
TCP/IP 协议集 (TCP/IP protocol suite)	1135
局域网基准 (参考) 模型 (local area network reference model)	439
网络协议工程 (network protocol engineering)	849
网络协议形式描述技术 (network protocol formal description technology)	850
网络协议一致性测试 (network protocol conformation testing)	851
网络计算模式 (network computing mode)	841
分时共享模式 (time-sharing mode)	187
资源共享模式 (resource-sharing mode)	1065
客户-服务器模式 (client/server mode)	479
浏览器-万维网-数据库模式 (browser-Web-database mode)	509
网络运行环境 (network operation environment)	852
网络操作系统 (network operating system, NOS)	832
网络数据库 (network database)	846
网络中间件 (network middleware)	853
分布式计算环境 (distributed computing environment)	180
网络服务质量 (quality of network services)	835
网络集成服务 (network integrated services)	841
网络区分服务 (network differential services, diffserv, DS)	841
网络资源预约协议 (network resource reservation protocol, RSVP)	853
局域网 (local network)	437
局域网拓扑结构 (topology of local area network)	441
局域网介质访问控制方法 (medium access control method of local area network)	439
局域网基准 (参考) 模型 (local area network reference model)	439

局域网逻辑链路控制子层(logical link control sublayer of local area network)	440
局域网介质访问控制子层(media access control sublayer of local area network)	440
局域网物理介质无关子层(physical medium-independent sublayer of local area network)	442
局域网物理介质相关子层(Physical medium - dependent sublayer of local area network, PMD)	443
局域网协议标准 (Protocol Standards of Local Area Network—IEEE 802 Std)	443
低速局域网(traditional local area network)	
以太网(Ethernet)	953
令牌环网(token ring network)	576
令牌总线网(token bus network)	577
专用小交换机(private branch exchange, PBX)	1063
高速局域网(high speed local area network)	209
光纤分布数据接口(fiber distributed data interface, FDDI)	246
快速以太网(fast Ethernet)	484
千兆位以太网(Gigabit Ethernet)	565
异步传送模式局域网(asynchronous transfer mode local area network, ATM LAN)	957
无线局域网(wireless local area network)	878
广域网(wide area network, WAN)	250
交换技术(switching technologies)	426
电路交换(circuit switching)	118
存储转发交换(store and forward switching)	99
分组交换(packet switching)	188
帧中继交换(frame relay switching)	1015
信元交换(cell switching)	912
公用交换电话网(public switched telephone network, PSTN)	223
专用小交换机(private branch exchange, PBX)	1063
专线网(dedicated line network)	
租用线路网(leased line network)	1083
虚拟专网(virtual private network, VPN)	926
数字数据网(digital data network, DDN)	743
公用数据网(public data network, PDN)	223
综合业务数字网(integrated services digital network, ISDN)	1079
异步传送模式(asynchronous transfer mode, ATM)	955
同步光纤网(synchronous optical network, SONET)	781

点对点连接协议 (point-to-point protocol, PPP)	115
移动通信网 (Mobile Communication Network)	949
全球移动通信系统 (Global System for Mobile Communication, GSM)	573
无线应用协议 (wireless application protocol, WAP)	880
个人通信网 (personal communication network, PCN)	216
卫星通信 (satellite communication)	866
甚小口径天线地球站 (very small aperture terminal, VSAT)	652
低轨道卫星通信系统 (low earth orbit communication system)	112
城域网 (metropolitan area network)	61
交换式多兆位数据业务 (switched multi-megabit data service, SMDS)	426
Internet (因特网)	1111
Internet 体系结构 (Internet architecture)	1114
TCP/IP 协议集 (TCP/IP protocol suite)	1135
网际协议 (internet protocol, IP)	830
传输控制协议 (transmission control protocol, TCP)	80
用户数据报协议 (user datagram protocol, UDP)	978
新一代网际协议 (new generation internet protocol, IPv6)	907
Internet 地址 (Internet address)	1112
域名系统 (domain name system, DNS)	993
Internet 基本服务 (basic services of Internet)	1113
虚拟终端 (virtual terminal, VT)	925
文件传送 (file transfer)	872
电子邮件 (electronic mail, E-mail)	130
Internet 信息服务 (Internet information services)	
万维网 (world wide web, WWW)	826
主页 (home page)	1057
浏览器 (browser)	509
Java 语言 (Java language)	1116
Gopher 服务 (Gopher Services)	1108
广域信息服务系统 (wide area information server, WAIS)	251
USENET 新闻 (USENET news)	1139
Listserv 论坛 (Listserv)	1120
Internet (因特网) 服务提供者 (Internet service provider, ISP)	1112
内联网 (Intranet)	544
外联网 (Extranet)	826
网络互连 (internetworking)	839
网络互连技术 (internetworking techniques)	839
网络互连协议 (internetworking protocol)	840

网际协议(internet protocol, IP)	830
X. 75 建议(X. 75 Recommendation)	1145
路由选择(routing)	512
内部路由协议(interior routing protocol)	544
OSPF 协议(Open Shortest Path First Protocol)	1127
RIP 协议(Routing Information Protocol)	1132
外部路由协议(exterior routing protocol)	822
边界网关协议(Border Gateway Protocol, BGP)	11
广播路由选择(broadcast routing)	250
网络互连设备(internetworking equipment)	840
中继器(repeater)	1048
网桥(bridge)	854
交换机(switch)	425
路由器(router)	511
网关(gateway)	830
网络管理(network management)	836
OSI 管理体系结构(OSI management architecture)	1126
集中式网络管理(centralized network management)	334
分布式网络管理(distributed network management)	183
网络管理标准(Network Management Standard)	837
公共管理信息协议(Common Management Information Protocol, CMIP)	221
管理信息库(management information base, MIB)	236
远程网络监控(remote network monitoring, RMON)	999
简单网络管理协议(Simple Network Management Protocol, SNMP)	419
电信管理网络(Telecommunication Management Network, TMN)	119
网络安全(network security)	831
密码学(cryptology)	523
私钥密码技术(private key cryptography)	757
公钥密码技术(public key cryptography)	224
鉴别(authentication)	421
Kerberos 鉴别(Kerberos authentication)	1117
数字签名(digital signatures)	741
公钥基础设施(public key infrastructure, PKI)	224
访问控制(access control)	164
防火墙(firewall)	161
虚拟专网(virtual private network, VPN)	926
计算机病毒(computer virus)	342
网络入侵检测(network intrusion detection)	842

安全服务(security services)	1
安全机制(security mechanisms)	1
信息系统安全评估准则(security evaluation criteria of information system)	912
可信计算机系统评估准则(trusted computer system evaluation criteria, TCSEC)	477
计算机信息系统安全保护等级划分准则(classified criteria for security protection of computer information system)	391
网络工程(network engineering)	836
网络规划(network planning)	838
网络设计(network design)	843
网络测试(network test)	832
传输介质(transmission medium)	
双绞线电缆(Twisted pair cable)	751
同轴电缆(coaxial cable)	782
光缆(Fiber optical cable)	242
结构化布线系统(structured cabling system, SCS)	428
工作区子系统(work area subsystem)	219
水平子系统(horizontal subsystem)	752
干线子系统(backbone subsystem)	202
设备间子系统(equipment subsystem)	645
管理子系统(administration subsystem)	238
建筑群子系统(campus subsystem)	420
布线系统标准(cabling system standard)	39
计算机网络应用(computer network application)	
网络应用服务(network application service)	852
文件共享(file sharing)	874
消息处理系统(message handling system, MHS)	901
公告板系统(bulletin board system, BBS)	221
附网存储(network attached storage, NAS)	194
计算机支持的协同工作(computer supported cooperative work, CSCW)	404
群件(groupware)	578
远程教育(distance education, Distance learning)	998
远程医疗(Telemedicine)	1000
电子商务(electronic commerce, EC)	124
电子数据交换(electronic data interchange, EDI)	128
电子商务标准(the standard for electronic commerce)	124
电子政务(electronic government)	131
办公自动化(office automation)	3

数字图书馆(digital library)	743
计算机集成制造系统(computer integrated manufacturing system, CIMS)	360
指挥控制通信情报系统(command, control, communication and intelligence system)	1037
持续采办和全寿命支持(continuous acquisition and life-cycle support, CALS)	73

VI. 计算机应用技术

计算机应用技术(technology for computer application)	395
中文信息处理(Chinese information processing)	1050
汉字(Hanzi, Chinese character, Han character, Chinese Hanzi)	267
汉字字汇(Ideographic repertoire)	277
汉字字型(Hanzi font, Hanzi typeface)	277
汉字字序(Hanzi order)	278
中日韩统一汉字(Unified CJK Ideographs)	1049
汉字编码字符集(coded Chinese character set (狭义), coded Ideographic character set (广义))	269
信息交换用汉字编码字符集 GB 2312-1980(Chinese Ideograms Coded Characters Set for Information Interchange-Basic Set, GB 2312-1980)	910
汉字内码扩展规范 GBK(GBK Chinese Internal Code Extension Specification)	272
信息交换用汉字编码字符集的扩充 GB 18030(Chinese Ideograms Coded Characters Set for Information Interchange-Extension for the Basic Set, GB 18030,)	911
通用多八位编码字符集 ISO/IEC 10646 (Universal Multiple-Octet Coded Character Set-ISO/IEC 10646, UCS)	779
Unicode 编码字符集(Unicode Coded Character Set)	1137
汉字键盘输入(Chinese character input via keyboard)	271
汉字编码(键盘)输入方法(Chinese character coding(keyboard) input method)	269
汉语拼音输入法(Chinese pinyin input method)	263
汉字部件(字根)输入法(Chinese character component input method)	271
汉字笔画输入法(Chinese character stroke input method)	268
汉字音形输入法(Chinese character phonological and calligraphic input method)	276
汉字形音输入法(Chinese character calligraphic and phonological input method)	276

汉语言语(语音)信号处理(Chinese speech signal processing)	
汉语言语(语音)识别(Chinese speech recognition)	265
言语(语音)识别的特征抽取(feature extraction of speech recognition)	935
汉语言语(语音)识别分类(classification of Chinese speech recognition)	267
言语(语音)识别中的语言模型(language model of speech recognition)	936
汉语言语(语音)理解(Chinese speech understanding)	264
汉语言语(语音)合成*(speech synthesis of Chinese)	264
言语(语音)合成方法(method of speech synthesis)	934
文语转换(text to speech, TTS)	875
汉字识别(Chinese character recognition)	273
汉字识别基本方法(basic method of Chinese character recognition)	275
汉字识别特征(features for Chinese character recognition)	276
印刷文本版面分析(printed page layout analysis)	961
联机手写汉字识别(online handwritten Chinese characters recognition)	505
印刷体汉字识别(printed Hanzi recognition)	961
脱机手写汉字识别(off-line handwritten Chinese characters recognition)	820
汉字识别后处理(Chinese characters recognition postprocessing)	274
机器翻译(machine translation, MT)	296
自然语言处理(natural language processing, NLP)	1074
中文信息检索(Chinese information retrieval)	1050
信息检索方法(information retrieval method)	909
自动标引(automatic indexing)	1069
全文检索(full-text retrieval)	575
信息检索中的相关反馈(relevance feedback in information retrieval)	909
语料库(corpus)	986
电子印前处理(electronic pre-press processing)	129
排版软件(page layout software)	547
页面描述语言(page description language)	944
印前图像处理技术(prepress image processing technology)	960
激光照排机(laser beam image typesetter)	315
计算机图形学(computer graphics)	379
图元生成(graphics primitive generation)	818
图形变换(graphics transformation)	817
图形裁剪(graphics clipping)	817
曲线(curve)	571

参数曲线(parametric curve)	41
曲面(surface)	570
参数曲面(parametric surface)	40
隐函数曲面(implicit function surface)	959
剖分曲面(subdivision surface)	560
造型技术(modeling technique)	1008
几何造型(geometric modeling)	335
自然景物造型(modeling of natural phenomena)	1073
基于物理的造型(physically-based modeling)	314
多分辨率造型(multi-resolution modeling)	142
基于图像的造型(image-based modeling, IBM)	313
计算机图形标准(computer graphics standard)	378
GKS 图形标准(Graphical Kernel System Graphics Standard)	1108
PHIGS 图形标准(Programmer's Hierarchical Interactive Graphics System, PHIGS)	1130
CGI 图形接口标准(Computer Graphics Interface Standard, CGI)	1101
图形元文件标准(Computer Graphics Metafile Standard, CGM)	818
OpenGL 图形标准(OpenGL Graphics Standard)	1126
颜色处理(color processing)	
颜色模型(color model)	938
颜色复制(color reproduction)	936
颜色管理(color management)	937
真实感图形生成(realistic graphics generation)	1011
消隐技术(hidden line/surface removal techniques)	903
纹理生成(texture generation)	876
纹理映射(texture mapping)	876
光照模型(illumination model)	249
光亮度计算(luminance calculation)	243
Phong 明暗处理(Phong shading)	1131
Gouraud 明暗处理(Gouraud shading)	1109
光线跟踪技术(ray tracing technique)	247
辐射度技术(radiosity technique)	193
图形反走样技术(anti-aliasing technique)	817
基于图像的绘制(image-based rendering, IBR)	312
基于全视函数的绘制(plenoptic function based rendering)	311
视图插值(view interpolation)	668
实时绘制(real time rendering)	662
非真实感图形绘制(non-photorealistic rendering, NPR)	175

计算机动画 (computer animation)	345
卡通动画 (cartoon animation)	456
三维动画 (3D animation)	638
关键帧动画 (keyframe animation)	233
算法动画 (algorithmic animation)	761
基于物理的动画 (physically based animation)	313
变形体动画 (soft object animation)	14
人体动画 (human body animation)	592
表情动画 (facial expression animation)	17
动画描述语言 (animation script language)	134
动画后期处理 (animation post-processing)	133
科学计算可视化 (visualization in scientific computing)	465
等值面构造技术 (iso-surfaces construction technique)	111
体绘制技术 (volume rendering technique)	770
三维空间数据场可视化 (visualization of data sets in 3D space)	638
虚拟现实 (virtual reality, VR)	922
虚拟现实输出设备 (output devices of virtual reality)	923
虚拟现实输入设备 (input devices of virtual reality)	924
增强现实 (augmented reality, AR)	1008
数字几何处理 (digital geometry processing)	735
三维网格模型简化 (simplification for 3D mesh model)	640
三维网格模型参数化 (parameterization for 3D mesh model)	639
三维网格处理 (3D mesh processing)	639
数字图像处理 (digital image processing)	744
图像模型 (image model)	806
图像获取 (image acquisition)	801
图像表示 (image representation)	792
图像边界表示 (image boundary representation)	788
图像区域表示 (image region representation)	808
图像几何特征表示 (image geometric feature representation)	802
图像矩表示 (image moment representation)	804
图像骨架表示 (image skeleton representation)	801
图像变换 (image transformation)	790
图像处理的基本运算 (basic operations in image processing)	795
图像点运算 (image point operation)	798
图像邻域运算 (image neighborhood operation)	806
图像变换运算 (image transform operation)	792
图像几何运算 (image geometric operation)	803

图像形态学运算(image morphological operation)	814
图像增强(image enhancement)	816
图像复原和重建(image restoration and reconstruction)	
图像退化(image degradation)	811
图像复原(image restoration)	800
图像反向滤波复原(image restoration by inverse filtering)	799
图像维纳滤波复原(image restoration by wiener filtering)	812
图像运动模糊复原(image motion-blurred restoration)	815
图像重建(image reconstruction)	794
图像分割(image segmentation)	800
图像像素分类(image pixel classification)	813
图像区域分割(image region segmentation)	809
图像边缘检测(image edge detection)	789
图像特征提取(image feature extraction)	809
图像细化(image thinning)	813
图像的压缩编码(compression and coding of images)	796
图像并行处理(image parallel processing)	793
多媒体技术(multimedia technology)	144
数字音频处理(digital audio processing)	
数字音频获取(acquisition of digital audio)	749
数字语音的压缩编码(compression and coding of digital voice)	732
全频带数字音频的编码(coding of full band digital audio)	572
数字音频编辑(editing digital audio)	749
计算机音乐(computer music)	394
数字视频处理(digital video processing)	
数字视频获取(acquisition of digital video)	742
静止图像的压缩编码标准(compression and coding standards of still images)	437
运动图像的压缩编码标准(compression and coding standards for motion image)	1003
数字视频编辑(editing digital video)	742
视频光盘机(video compact disc player)	665
数字多用途光盘(digital versatile disc,DVD)	730
多媒体文档(multimedia document)	146
超文本(hypertext)	59
多媒体文档规范(specifications of multimedia document)	147
多媒体著作工具(multimedia authoring tool)	148
分布式多媒体系统(distributed multimedia system)	179

计算机支持的协同工作(computer supported cooperative work, CSCW)	404
交互式电视(interactive television)	424
可视电话(videophone)	477
视频会议(videoconferencing)	666
流媒体(streaming media)	510
远程教育(distance education, distance learning)	998
远程医疗(Telemedicine)	1000
人机交互技术(human-computer interaction technology, or HCI technology)	588
多模态人机交互(multimodal human-computer interaction)	148
生物特征识别(biometrics)	654
说话人识别(speaker recognition)	755
指纹识别(Fingerprint Recognition)	1042
人脸识别(face recognition)	591
虹膜识别(iris recognition)	280
手势识别(hand gestures recognition)	670
多模态生物特征融合(multi-modal fusion in biometrics)	149
计算机游戏(computer game)	401
计算机信息系统(computer information system)	390
数据库管理(database administration)	691
决策支持系统(decision support system, DSS)	452
联机分析处理(on-line analytical processing, OLAP)	503
数据仓库(data warehouse)	681
数据挖掘(data mining)	706
联机事务处理(online transaction processing, OLTP)	505
管理信息系统(management information system, MIS)	237
电子商务(electronic commerce, EC)	124
电子政务(electronic government)	131
地理信息系统(geographic information system, GIS)	113
遥感信息处理(remote sensing information processing)	943
全球定位系统(global positioning system, GPS)	573
办公自动化(office automation)	3
数字图书馆(digital library)	743
军事指挥信息系统(military command information system)	454
计算机辅助技术(technology for computer aided something)	350
计算机辅助绘图(computer aided drafting)	349
计算机辅助设计(computer aided design, CAD)	352
计算机辅助制造(computer aided manufacturing, CAM)	353
计算机辅助工艺规划(computer aided process planning, CAPP)	348

柔性制造系统(flexible manufacturing system, FMS)	598
计算机辅助质量控制(computer aided quality control, CAQ)	354
计算机集成制造系统(computer integrated manufacturing system, CIMS)	360
并行工程(concurrent engineering)	26
企业资源规划(enterprise resource planning, ERP)	562
产品数据管理(product data management, PDM)	50
企业流程重组(business process reengineering, BPR)	562
工程数据库(engineering database)	217
产品数据交换标准(Product Data Exchange Specification, PDES)	51
电子设计自动化(electronic design automation, EDA)	126
硬件描述语言(hardware description language, HDL)	973
数字系统逻辑综合技术(logic synthesis technologies for digital system)	746
数字系统模拟技术(simulation technologies for digital system)	747
布图技术(layout technology)	38
计算机辅助教学(computer assisted instruction, CAI)	351
课件(courseware)	479
计算机控制(computer control)	363
计算机过程控制(computerized process control)	357
计算机过程控制方式(computerized process control manner)	358
顺序控制(sequential control)	754
工业控制计算机(industrial control computer)	219
单回路数字控制器(single loop digital controller)	109
可编程控制器(programmable controller, PC)	466
集散控制系统(distributed control system, DCS)	332
现场总线控制系统(fieldbus control system, FCS)	892
计算机仿真(computer simulation)	346
连续系统仿真(continuous system simulation)	502
离散事件系统仿真(discrete event system simulation)	493
离散事件系统仿真建模方法学(modeling methodology of discrete event system simulation)	495
离散事件系统仿真输出分析(output analysis of discrete event system simulation)	497
并行仿真(parallel simulation)	25
分布交互式仿真(distributed interactive simulation, DIS)	176
仿真语言(simulation language)	163
训练仿真器(training simulator)	932

VII. 人工智能

人工智能 (artificial intelligence)	586
知识工程 (knowledge engineering)	1020
知识表示 (knowledge representation)	1017
产生式表示 (production representation)	52
语义网络表示 (semantic network representation)	992
框架表示 (frame representation)	487
逻辑表示 (logic representation)	514
直接表示 (direct representation)	1027
常识 (commonsense)	54
信念 (belief)	908
陈述性表示 (declarative representation)	60
过程表示 (procedure representation)	257
自动推理 (automated reasoning)	1072
定理机器证明 (mechanical theorem proving)	132
归结方法 (resolution method)	251
自然演绎法 (natural deduction method)	1074
吴方法 (Wu method)	877
重写规则法 (rewriting rule method)	73
定性推理 (qualitative reasoning)	133
非单调推理 (non-monotonic reasoning)	168
约束推理 (constraint reasoning)	1002
模糊推理 (fuzzy reasoning)	531
类比推理 (analogical reasoning)	489
归纳推理 (inductive inference)	252
基于案例的推理 (case-based reasoning, CBR)	308
专家系统 (expert system)	1058
知识库 (knowledge base)	1026
推理机 (inference engine)	819
解释机制 (explanation mechanism)	431
专家系统开发环境 (expert system development environment)	1060
分布式专家系统 (distributed expert system, DES)	184
连接专家系统 (connectionist expert system)	500
知识获取 (knowledge acquisition)	1021
知识精化 (knowledge refinement)	1025
知识获取工具 (knowledge acquisition tool)	1023

基于内容的检索 (content-based retrieval)	311
基于 Agent 的计算 (agent-based computing)	307
启发式搜索 (heuristic search)	563
A* 算法 (A* algorithm)	1095
AO* 算法 (AO* algorithm)	1098
α - β 剪枝 (alpha-beta pruning)	1147
博弈树搜索 (game tree search)	34
局部搜索算法 (local search algorithm)	437
机器学习 (machine learning)	302
归纳学习 (inductive learning)	253
类比学习 (learning by analogy)	490
分析学习 (analytic learning)	187
数据挖掘 (data mining)	706
遗传学习 (genetic learning)	952
学习计算理论 (learning computation theory)	930
模式识别 (pattern recognition)	537
模式识别方法 (method of pattern recognition)	
模式分类器 (pattern classifier)	536
特征选择 (feature selection)	770
非监督学习 (unsupervised learning)	173
句法模式识别方法 (syntactic pattern recognition method)	448
人工神经网络在模式识别中的应用 (application of artificial neural networks to pattern recognition)	584
计算机视觉 (computer vision)	374
视觉计算理论 (theory of vision computing)	664
图像分析 (image analysis)	800
三维形态恢复 (three dimensional shape recovering)	640
图像理解系统 (image understanding systems)	805
立体视觉 (stereo vision)	499
运动分析 (motion analysis)	1003
距离图像获取与分析 (range image acquisition and analysis)	451
计算机视觉中的结构光法 (structured light method in computer vision)	375
摄像机标定 (camera calibration)	648
自然语言理解 (natural language understanding, NLU)	1077
自然语言处理的词法分析 (morphological analysis of natural language processing)	1075
自然语言处理的句法分析 (syntactic analysis of natural language processing)	1076

语义分析 (semantic analysis)	991
语境分析 (context analysis)	984
计算语言学 (computational linguistics)	414
短语结构语法 (phrase structure grammar, PSG)	137
格语法 (case grammar)	215
基于合一的语法理论 (unification-based grammars)	310
语料库语言学 (corpus linguistics)	988
计量语言学 (quantitative linguistics)	336
机器翻译 (machine translation, MT)	296
人助机译 (human-aided machine translation, HMT)	593
机助人译 (machine-aided human translation, MAHT)	306
译前编辑 (pre-editing)	959
译后编辑 (post-editing)	959
电子词典 (electronic dictionary)	121
机器翻译评价 (machine translation evaluation)	298
智能机器人 (intelligent robot)	1043
机器人传感器 (robot sensor)	299
机器人运动规划 (robot motion planning)	301
机器人视觉 (robotic vision)	300
机器人控制 (robot control)	300
计算智能 (computing intelligence)	
神经计算 (neural computing)	651
人工神经网络 (artificial neural networks)	582
感知机 (perceptron)	203
反传学习 (back-propagation learning)	159
Hopfield 神经网络模型 (Hopfield neural network model)	1110
玻耳兹曼机 (Boltzmann machine)	33
适应谐振理论 (adaptive resonance theory, ART)	669
自组织映射模型 (self-organizing mapping model)	1078
联想记忆 (associative memory)	506
小脑网络模型 (cerebellar model articulation controller, CMAC)	904
遗传算法 (genetic algorithm)	949
人工生命 (artificial life)	586

A

Akeman hanshu

阿克曼函数 (Ackermann's function) 一种递归而非原始递归的函数(参见递归函数和原始递归函数)。

阿克曼函数定义如下:

$$\begin{cases} \psi(0, y) = y + 1 \\ \psi(x + 1, 0) = \psi(x, 1) \\ \psi(x + 1, y + 1) = \psi(x, \psi(x + 1, y)) \end{cases}$$

该函数定义中包含双重递归。当 $x > 0$ 时, $\psi(x, y)$ 的值均可通过“较早”值 $\psi(x_1, y_1)$ 来定义。这里, 或者 $x_1 < x$ 或者 $x_1 = x, y_1 < y$ 。这一点可直观地说明该函数可计算。

ψ 是递归函数的严格证明参见文献[1], ψ 为非原始递归函数的严格证明参见文献[2]。

参考文献

1. Cutland N. Computability, An Introduction to Recursive Function Theory. Cambridge: Cambridge University Press, 1980

2. Peter R. Recursive Functions. New York: Academic Press, 1967 (陈火旺 贵可荣)

anquan fuwu

安全服务 (security services) 开放互连系统的安全体系结构中为实现安全通信与访问提供的服务。**开放系统互连基准 (参考) 模型**的安全体系结构标准 ISO 7498-2 从体系结构的观点描述了 5 种可选的安全服务、8 项特定的**安全机制**以及 5 种普遍性的安全机制。5 种可选的安全服务是: **鉴别**、**访问控制**、**数据保密**、**数据完整性和防止否认**。

鉴别 即身份鉴别, 是授权控制的基础。必须做到准确无歧义地将对方辨别出来, 同时还应该提供双向的认证, 即互相证明自己的身份。

网络环境下的身份认证更加复杂, 主要是要考虑到验证身份的双方一般都是通过网络而非直接交互。大量的黑客随时随地都可能尝试向网络渗透, 截获合法用户口令并冒名顶替, 以合法身份入网。所以目前一般采用的是基于对称密钥加密或公开密

钥加密的方法(参见密码学), 采用高强度的密码技术来进行身份认证。比较著名的有 **Kerberos 鉴别**, 良好隐私电子邮件加密标准(PGP)等方法。

访问控制 即授权控制, 是控制不同用户对信息资源的访问权限。对授权控制的要求主要有:

(1) 一致性, 也就是对信息资源的控制没有歧义性, 各种定义之间不冲突。

(2) 统一性, 对所有信息资源进行集中管理, 安全政策统一贯彻。

(3) 要求有审计功能, 对所有授权有记录可以核查。

(4) 尽可能地提供细粒度的控制。目前很多系统的**访问控制**实际上还是在于 UNIX 文件系统的模式, 不能很好地满足安全需求, 需要进行扩充。

数据保密 数据保密主要依赖于数据加密技术, 目前加密技术主要有两大类: 一类是基于对称密钥加密的算法, 也称私钥算法; 另一类是基于非对称密钥的加密算法, 也称公钥算法。这些加密技术都已经达到一个很高的强度。

具体到加密手段, 一般分软件加密和硬件加密两种。软件加密成本低而且实用、灵活, 更换也方便; 硬件加密效率高, 本身安全性高, 可根据不同需要采用不同的方法。

密钥管理包括产生、分发、更换等, 是数据保密的重要一环。

数据完整性 数据完整性是指通过网上传输的数据应防止被修改、删除、插入、替换或重发, 以保证合法用户接收和使用该数据的真实性。

防止否认 接收方要求对方保证不能否认收到的信息是发送方发出的信息, 而不是被他人冒名、篡改过的信息。发送方也会要求对方不能否认已经收妥的信息。防止否认对金融电子化系统很重要, **数字签名**是防止抵赖、防止否认的主要手段, 它给仲裁提供证据。

(胡道元)

anquan jizhi

安全机制 (security mechanisms) 开放互连系统的安全体系结构中为实现安全通信与访问提供

的机制。**开放系统互连基准(参考)模型**的安全体系结构标准 ISO 7498-2 从体系结构的观点描述了 5 种可选的**安全服务**、8 项特定的**安全机制**以及 5 种普遍性的安全机制。8 项特定的安全机制是: 加密机制、**数字签名机制**、**访问控制机制**、**数据完整性机制**、**鉴别交换机制**、**通信业务填充机制**、**路由控制机制**和**公证机制**。

加密机制 加密能为数据提供机密性, 保护通信信息流的有关信息。它能单独作为一种机制运行, 也能作为后面其他机制的一部分, 起到补充的作用。

数字签名机制 该机制的实质在于对特定数据单元的签名, 并且只有从形成该签名的那个机密信息中产生出来, 机密信息的持有者惟一。因此, 当该签名得到检验之后, 能够在任何时候向第三方即仲裁提供证明签名人的证据。

访问控制机制 该机制包括基于规则策略的访问控制方式和基于存取身份的访问控制方式两种。

数据完整性机制 数据完整性有两个方面, 单个数据单元或字段的完整性和单元流或字段流的完整性。

鉴别交换机制 鉴别交换机制设置在网络适当层次上提供对等实体**鉴别**。在鉴别实体时得到否定结果, 会导致连接的拒绝或终止, 可能在安全审计跟踪中增加一个记录, 也可能向安全管理中心作出报告。

鉴别交换机制使用由发送实体提供而由接收实体核验的鉴别信息(口令等), 使用密码技术或者使用该实体的特征。

通信业务填充机制 通信业务填充机制能用来提供各种不同级别的保护, 抵抗通信业务分析。它只有在通信业务受到保密服务保护时才有效。

路由控制机制 该机制保护通信路由能够动态地或预定地选取, 以便选用物理上的子网络、中继站或链路。

公证机制 公证机制是在多个实体之间交换数据信息时, 用于保护各个实体的安全以及纠纷的仲裁。

以上 8 项特定的安全机制是为了特定的**安全服务**所设置的。除此之外, 还有以下 5 种普遍性的安全机制, 虽然不是为特定的安全服务而设置, 但是与系统所要求的安全级别直接有关。

可信功能 任何功能, 只要它是直接提供安全机制, 或提供对安全机制的访问都应该是可信的。

安全标记 包含数据项的资源可以具有同该数据相关的安全标记, 如指明数据敏感性级别的标记等。它们在传送中必须同数据一起传送。

事件检测 与安全有关的检测包括检测对安全的明显侵害(如特定的安全攻击、特别选择的事件以及事件发生计数的溢出等检测), 也可能包括对普通“正常”事件的检测(例如一次成功的访问)。

安全审计跟踪 安全审计就是对系统的记录与行为进行独立的评审考查, 其潜在价值在于经过事后的审计能够检测和调查安全的漏洞。

安全恢复 安全恢复处理来自于诸如事件处理与管理功能等机制的请求, 并把恢复动作作为使用一组规则之后的结果。
(胡道元)

B

Bakesi fanshi

巴克斯范式 (Backus normal form, BNF)

用以精确描述程序设计语言语法的一种形式体系。又称巴克斯-诺尔形式或巴克斯-诺尔形式体系。

美国 IBM 公司研究员 J. Backus 和丹麦哥本哈根大学教授 P. Naur 最早用以描述 ALGOL 60 语言的局部语法(其全程语法并不能用 BNF 描述),因而得名。

相对 BNF 所描述语法之语言说来,BNF 为元语言,它所描述语法之语言为对象语言。

BNF 之基本成分为如下部分。

元符号,一个是元等价符“ $::=$ ”,表示“定义为”;一个是元或符“ $|$ ”,表示对所指出的各款项之选择;还有一对尖括号“ $\langle \rangle$ ”,用以连同所括之名表示非终极符。

元变量,用以表示相应对象语言之语法单位。元变量本身不属对象语言,其值为对象语言中之符号串。

以上述元符号及元变量为基础所构成的一组元公式(或称产生规则),用以刻画相应对象语言之局部语法。其形为

非终极符 $::=$ 非终极符与终极符组成之符号串
其中非终极符(即上述之元变量)刻画对象语言之语法单位,终极符为对象语言之符号串。

元公式中“ $::=$ ”之左方称为公式之左部,“ $::=$ ”之右方称为公式之右部,具有相同左部之不同公式可以公用同一个左部。

例如,在 ALGOL 60 中,有

$\langle \text{二进制数字} \rangle ::= 011$

$\langle \text{十进制数字} \rangle ::= 0111213141516171819$

$\langle \text{数字} \rangle ::= \langle \text{二进制数字} \rangle | \langle \text{十进制数字} \rangle$

元公式之右部亦可出现正在定义的左方非终极符,此时为递归定义。如

$\langle \text{标识符} \rangle ::= \langle \text{字母} \rangle | \langle \text{标识符} \rangle \langle \text{字母} \rangle | \langle \text{标识符} \rangle \langle \text{数字} \rangle$

即,标识符定义为以字母起头的字母数字串。

BNF 能严格表示一类上下文无关语言之局部语法,自从在 ALGOL 60 语言文本中首先采用以来,

已在不少计算机学科中得到广泛应用和推广。

BNF 一词有其演变过程,起初代表 Backus normal form 或 Backus-Naur form,后来人们认为,既然 BNF 为一形式体系,为一元语言,称之为 form,似觉欠妥,F 以代表 formalism 为佳,从而便出现 Backus-Naur formalism,这里 formalism 一词与 formal system 同义。

参考文献

1. Naur P et al. Report on the Algorithmic Language ALGOL 60. CACM, Vol. 3, No. 5, May 1960, 299 ~ 314
2. 陈火旺等. 程序设计语言编译原理. 北京: 国防工业出版社, 1984 (陈火旺 程虎 贵可荣)

Bakesi-Nuor Xingshi tixi

巴克斯-诺尔形式体系 (Backus-Naur formalism, BNF) 参见巴科斯范式。

bangong zidonghua

办公自动化 (office automation) 利用现代化办公设备、计算机技术和通信技术,代替手工作业,以提高办公效率的一种管理手段。这一管理手段涉及行为科学、系统科学、计算机技术和通信技术的综合应用。随着电子商务、电子政务的发展,办公自动化在公共部门、企业以及其他机构得到普遍应用。

一个单位或部门的办公自动化应用涵盖其日常办公事务的全部相关事宜,其中主要的内容包括:

(1) 公文处理 实现公文的签收、登记、分发、归档、会签和发文。

(2) 电子邮件 依据不同工作流的处理流程,自动地将文档以电子邮件的方式传递给下一处理部门或人员。

(3) 签报管理 对部门内部形成和使用的特殊文件进行管理,涉及拟稿、审核、签收、会签、拟办、签批、电子签名、交办、退稿、备查、提高归档、打印、销毁等。

(4) 会议会务管理 包括常规会议和临时会议

管理、提交计划准备、记录、查询等功能。

(5) 信息资料管理 对原始形式的数据资料进行管理,具有编辑、文摘与提要编写、刊号管理,以及查询、统计、汇编等功能。

(6) 案卷管理 将相关文件编辑成卷,提供查询案卷功能,对借阅过程进行控制。

(7) 领导办公。

(8) 公共信息服务 主要包括机构设置、职能与人员编制简介,业务介绍、通讯录、公告板、系统使用方法等。

(9) 专题电子论坛。

(10) 系统管理 管理和维护办公自动化系统、管理机构、人员、应用数据库等方面的公共信息,以达到配置系统和改变工作流,适应各种不同应用环境的目的。通过对机构、人员、应用数据库等公共信息进行集中、统一的管理和维护,保证综合办公系统公共信息的完整性、统一性和独立性。

一般认为,办公自动化系统由6个要素组成,即办公人员、组织机构、办公制度、技术工具、办公信息和办公环境。办公自动化系统实质上是一个信息系统,可辅助办公人员进行办公信息处理和决策。

办公自动化运行环境的特点有:①地理位置分散;②资源共享,功能分布;③有时需要移动办公;④需要 workflow 管理机制支持,协同工作;⑤具有异构集成性。由此,一个实用高效的办公自动化系统,应是一个具有多结点的交互人机会话网络系统,各办公部门能承担相对独立的任务,并可相互配合,具有协同工作能力;办公信息系统同时又是一个分布

多用户的数据库系统,不同用户既有共享办公资源的能力,又有不同的数据视图与不同的数据访问权限。

办公自动化一词首创于1936年,意即运用打字机、电话设备帮助办公人员处理办公业务。随着现代计算机和网络通信技术的发展,办公自动化的概念无论是外延还是内涵,都发生了很大变化。一般认为办公自动化经历了3个阶段:①以数据处理为中心的阶段,表现为以个人计算机、办公套件为特征,实现数据统计和文档写作电子化,使办公信息载体从纸介质方式向电子方式转换。②以网络为基础,以工作流为核心的阶段,特征表现在以电子邮件、文档管理、目录服务以及群组协同工作为特征。③建立以知识管理为核心的,能够在网上实时交流信息的集成平台,主要表现为在公文和档案管理等办公业务一体化自动处理的基础上,在办公处理的每一环节上提供相关知识,使办公人员在系统中的地位从被动向主动转变,在提升办公人员创造能力的过程中,提高单位或部门整体办公质量、效率和应变能力。

(孙志挥)

bandaoti cunchuqi

半导体存储器 (semiconductor memory)

用半导体大规模集成存储器芯片作为存储介质并能对数字信息进行存取的存储设备。典型半导体存储器的基本组成如图1所示,各部分的功能简述如下。

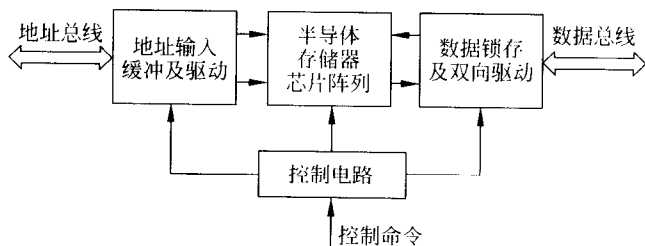


图1 典型半导体存储器的基本组成框图

(1) 半导体存储器芯片阵列 它是信息存储单元的集合,由半导体存储器芯片按一定结构组成,是供中央处理器(CPU)或输入输出设备(I/O)存取信息的存储空间。

(2) 地址输入缓冲及驱动 由地址输入寄存器、片选译码器、地址分时电路和地址驱动器等几部

分组成。地址输入寄存器的功能是接收CPU给出的地址,再将低位地址直接送给存储器芯片阵列,而高位地址经过译码电路产生片选或块选信号。刷新计数器和地址分时电路为动态随机存取存储器(DRAM)所必需,它们的功能是将器件输入的行地址和列地址归并为一,并将所产生的刷新地址送

至存储器芯片的地址端口。

(3) 数据锁存及双向驱动 包含输入缓冲门、输入数据寄存器、输出锁存器和输出缓冲门等。它的功能是根据 CPU 的读写命令,将数据总线的内容写入被访问的存储单元,或者将被访问存储单元的内容读出到数据总线上,供 CPU 或 I/O 使用。

(4) 控制电路 包括访问信号控制和刷新电路。它的功能是接收 CPU 的启动、读、写、清除等命令,并对接收后的命令进行处理,产生各种时序控制脉冲和内部定时脉冲信号来控制存储器的读写操作或刷新操作。

在半导体存储器的组成中,存储器芯片阵列是核心,其余三部分是存储器的外围电路。

半导体存储器按在计算机中的工作性质可分为**主存储器(内存)**、**辅助存储器(外存)**、**高速缓冲存储器**和控制存储器等。主存储器具有速度高、存储容量较小的特点,用来存放 CPU 当前执行的程序和数据。通常主存储器是由随机存取方式和只读方式两种半导体存储器组成,它们分享主存储器的同一地址空间。辅助存储器的特点是速度慢,存储容量大,用来存放 CPU 暂时不用的程序和数据。磁盘是辅助存储器的主流,但半导体盘(**固态硬盘**)已能实现磁盘的功能,并已在一些应用领域取代了磁盘。从功能上看半导体盘也可由电荷耦合器件(CCD)存储器实现,但从容量、速度和成本等因素综合考虑,则宜采用由 MOS 动态存储器(DRAM)或快可擦编程只读存储器构成。缓冲存储器用于在两个不同工作速度的部件之间交换信息过程中起缓冲作用。高速缓冲存储器是一种容量较小、速度很高的存储器,是为了克服主存储器与 CPU 在数据传输速度上的不匹配而设置的,通常采用双极型半导体存储器芯片来制作。磁盘缓冲存储器是为克服主存储器与磁盘存储器之间数据传输速度的不匹配而设置,利用磁盘缓存的方法可缩短磁盘平均取数时间,进而提高磁盘系统使用效率。磁盘缓冲存储器通常用 MOS DRAM 构成。

半导体存储器芯片种类很多,就其制造工艺而言可分为**双极型半导体存储器**和**金属-氧化物-半导体存储器**(简称 MOS 存储器)。双极型半导体存储器以双极型触发器作为存储单元。其中采用晶体管-晶体管逻辑存储单元的称为 TTL 型存储器,采用射极耦合逻辑存储单元的称为 ECL 型存储器。MOS 存储器以金属-氧化物-半导体场效应晶体管作为存储单元,按工作方式不同分为静态随机存取存

储器(SRAM)和动态随机存取存储器(DRAM),MOS 存储器的特点是集成度高,工作速度较快,适用于作容量较大的主存储器。电荷耦合器件(CCD)也是金属-氧化物-半导体存储器件,基于此器件构成的存储器称为**电荷耦合器件存储器**。它是一种易失性串行存储器,通常为**非破坏读出**,仅当写入新信息时,才清除原来存储的信息。

半导体存储器芯片按存取方式可分为**只读存储器芯片(ROM)**和**随机存取存储器芯片(RAM)**两大类。

RAM 可以随机地按指定地址从存储单元存取数据,在半导体存储器出现以前,主要是以记忆磁心为存储单元的磁心存储器。1971 年美国 Intel 公司推出 1103 型 1 kb MOS DRAM 芯片后,半导体存储器开始在一些主要厂家用作计算机的主存储器,到 1976 年 MOS RAM 主存储器的每信息位的价格已降到磁心存储器的一半,从此 MOS RAM 取代了磁心存储器,并一直占据主导地位。

只读存储器的基本特征是在正常运行中只能随机读取预先存入的信息而不能写入新的内容,也即信息一旦写入就不能更改。即使在断电情况下,ROM 仍能长期保存信息内容不变,所以它是一种永久存储器。

随着大规模集成电路集成技术的发展,出现了多种大规模集成电路 ROM 芯片,其中掩模只读存储器(mask ROM)结构简单,存储信息稳定,可靠性高,能够永久性保存信息。可编程只读存储器(PROM)是由半导体厂家制作“空白”存储阵列(即所有存储单元全部为 1,或全部为 0 状态)出售,用户根据需要可以实现现场编程写入,但只能实现一次编程。可擦编程只读存储器(EPROM)、电可擦编程只读存储器(EEPROM)和快可擦编程只读存储器(Flash EPROM)等 ROM 不仅可以现场编程,还可以擦除原存储的信息内容,写入新的信息。目前使用的 ROM 都是大规模集成电路存储器芯片,当它们装入程序或微程序后就称为**固件**。

半导体存储器自 1971 年以来发展迅猛。以 MOS RAM 为例,集成度平均以每三年增加 4 倍的速度增长,存取周期缩短,逻辑功能加强。在 MOS RAM 中,DRAM 的集成度一直约为 SRAM 的 4 倍,因此 DRAM 多用于大容量存储器中。由于 DRAM 存储单元利用电容存储电荷的机理保存信息,故使用 DRAM 时必须定时周期性刷新所存储的信息,以免丢失。SRAM 不需要刷新操作,使用简便,在一些

容量稍小的存储器中使用较广泛。由于 MOS SRAM 的速度不断提高,已几乎替代了双极型 RAM,促使双极型 TTL RAM 向更高速度方向发展。

半导体存储器由于存储单元阵列及其外围电路可集成在同一芯片上,因而生产过程简单,调试方便,容易实现自动化。它具有可靠性高,结构紧凑,组装密度高,体积小,工作速度快等特点;其缺点是信息易失性,所存信息因断电而消失。在不允许信息消失的应用场合,必须采取断电保护措施。但半导体存储器的综合优势是其他类型存储器不能相比的,在相当长时期内其性能与应用范围还将有较大发展。

参考文献

郑筠. MOS 存储系统及技术. 北京: 科学出版社, 1990

(郭志先)

bandaoti cunchuqi xinpian

半导体存储器芯片 (semiconductor memory chip) 用半导体集成电路制造工艺把很多存储单元电路排列成阵列, 并和外围电路集成在同一硅晶片上, 形成能储存大量数据的集成电路。它广泛用于计算机、通信和家用数字电器等数字系统。

每个存储单元可以存放一位二进制数, 称为一位。在数字系统中, 指令和数据(包括声音、图像等信息)都以二进制数的形式存放在存储单元阵列中。外围电路包括选取存储单元的地址缓冲和译码驱动电路、读出数据放大电路、写入数据驱动电路、片选和内部时序控制电路等。

由于半导体存储器(SM)芯片由大量相同的存储单元组成, 电路逻辑结构简单, 特别易于大规模集成, 而且 SM 芯片又是数字系统中非常重要的组成部分, 因此从 20 世纪 60 年代末以来, 随着半导体工艺的发展, SM 得到长足的进展, 成为半导体集成电路产品市场非常重要且不可缺少的部分。不仅有单独的 SM 芯片, 而且在系统集成的芯片上也集成了不同类型的存储器电路。

SM 芯片因制造工艺的不同, 可分为双极型、MOS 型和 CMOS/BiCMOS 型三类, 这也是 SM 芯片的发展历程。自 20 世纪 90 年代以来, CMOS/BiCMOS 几乎占领了 SM 芯片的全部市场。

根据断电后对数据的保存能力, SM 芯片又可分为易失性存储器芯片和非易失性存储器芯片两大

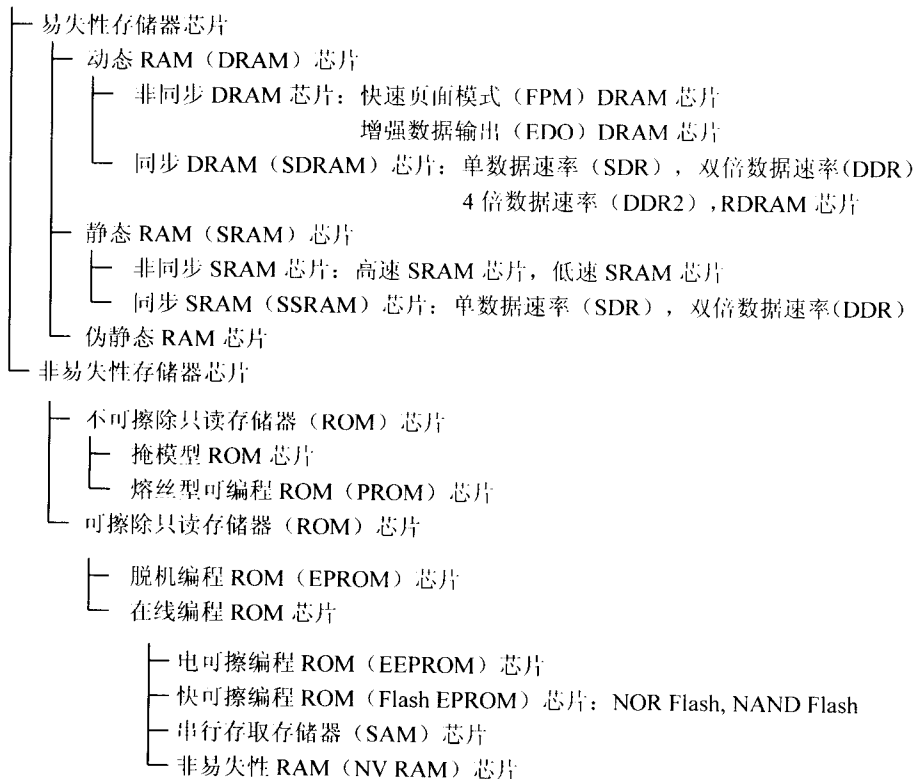
类。易失性存储器分为随机存取存储器与非随机存取存储器。非随机存取存储器主要有寄存器组、移位寄存器、先进先出(FIFO)存储器、后进先出(LIFO)存储器、按内容寻址存储器等。这些存储器现多以模块方式集成在芯片上, 不以单独芯片方式出现。所以易失性存储器芯片一般指随机存取存储器(RAM)芯片(参见**随机存取存储器芯片**)。它可从任何给定地址的存储单元快速读出或写入数据, 但断电后数据消失。非易失性存储器芯片在断电后所存储的数据仍能保存, 通常指只读存储器(ROM)芯片(参见**只读存储器芯片**)。比较细的 SM 芯片分类如表 1 所示。从表 1 中可看出: RAM 芯片有动态 RAM(DRAM)芯片(参见**动态随机存取存储器芯片**)、静态 RAM(SRAM)芯片(参见**静态随机存取存储器芯片**)和伪静态 RAM 芯片(参见**随机存取存储器芯片**)。DRAM 芯片必须周期性地对所存数据进行刷新, 以防止数据消失, 而 SRAM 芯片则不需要刷新。伪静态 RAM 芯片具有 DRAM 芯片的内核和 SRAM 芯片的接口。DRAM 芯片按接口的不同可分为同步 DRAM(SDRAM)芯片和非同步 DRAM 芯片。SDRAM 芯片有不同的数据传输速率。同样, SRAM 芯片也可分为同步 SRAM(SSRAM)芯片和非同步 SRAM 芯片。ROM 芯片分为不可擦除 ROM 芯片和可擦除 ROM 芯片两种。不可擦除 ROM 芯片包括掩模型 ROM 芯片和熔丝型可编程 ROM(PROM)芯片(参见**可编程只读存储器芯片**)。掩模型 ROM 芯片所存储的内容在芯片制造时靠掩模编码写入。PROM 芯片所存内容可由用户通过编程一次性地写入。可擦除 ROM 芯片可以多次擦除和编程写入, 包括可擦编程 ROM(EPROM)芯片(参见**可擦编程只读存储器芯片**)、电可擦编程只读存储器(EEPROM)芯片(参见**电可擦编程只读存储器芯片**)、快可擦编程只读存储器(Flash EPROM)芯片(参见**快可擦编程只读存储器芯片**)、串行存取存储器(SAM)芯片、非易失性 RAM(NV RAM)芯片(参见**只读存储器芯片**)等。EPROM 芯片用紫外线擦除整个芯片所存数据, 然后重新写入所需的数据。EEPROM 芯片用电的方法擦除。Flash EPROM 芯片也称**闪存芯片**, 是一种基于单管存储单元结构的 EEPROM 芯片, 因而集成度可以很高。SAM 芯片是在数据输入输出接口处有并串行转换的移位寄存器的 EEPROM 芯片, 主要用于掉电后希望保存少量数

据(例如系统配置参数和网络地址等)的数字系统。NV RAM 芯片是将 DRAM(或 SRAM)阵列和 EEPROM 阵列组合在同一芯片上。EEPROM 中可存放

系统配置参数。芯片加电时,EEPROM 中的内容自动写入 RAM。系统运行时,可通过 RAM 修改参数,并写入 EEPROM。

表 1 半导体存储器芯片分类

半导体存储器芯片



半导体存储器芯片因其存储容量大、体积小、功耗低、存取速度快以及便于和其他逻辑电路接口等优点,在数字系统中得到广泛应用。随着半导体工艺的发展,其集成度不断提高,存取速度不断加快,应用范围也不断扩大。满足特殊应用的专用半导体存储器芯片,例如,以 DDR/SDR SDRAM 为基础的图像存储器芯片、网络存储器芯片、移动通信(手机)的 SDRAM 芯片、以 NOR Flash 为内核存储 BIOS 固件端口 (FWH) 闪存芯片等也相继进入市场。2003 年年底市场上已可见 1 Gb、时钟频率为 133 MHz/166 MHz/200 MHz 的 DDR 266/333/400 SDRAM 芯片和取数时间为 50 ns 的 4 Gb NAND Flash 芯片。

参考文献

1. Rabaey J M. Digital Integrated Circuits: A

Design Perspective. Prentice Hall, 1996. 北京:清华大学出版社,1998(影印版)

2. <http://www.samsung/Products/Semiconductor>
(孙祖希)

bandaoti duxie cunchuqi

半导体读写存储器 (semiconductor read-write memory) 可在有限的时间间隔内存入或取出数据的半导体存储器。它用地址描述数据处在存储器的位置,通过所提供的地址码可随意地对任何存储单元进行写入或读出操作,其读周期和写周期是常数,与存储单元所处的位置无关。

半导体读写存储器应具备的基本功能部件包含存储器芯片阵列、地址缓冲和片选译码器、数据锁存和双向驱动器、时序发生器和控制器,并通过地址总

线、数据总线和控制总线与中央处理器(CPU)相连接,如图1所示。虽然半导体存储器芯片的集成度很高,但使用单个芯片作为主存储器,其容量还是不够的,必须在字向和位向两方面加以扩展,组成芯片阵列以满足存储容量的需要。

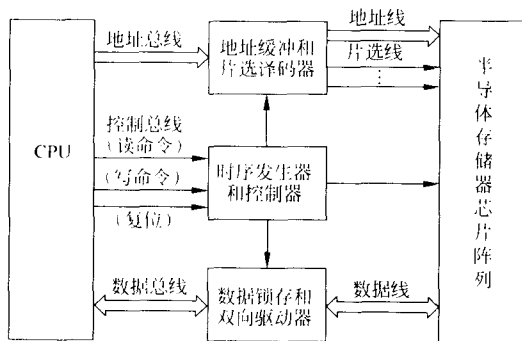


图1 半导体读写存储器组成框图

存储单元是数字计算机系统内一个信息单位的实际存储位置,信息单位的大小可以是二进制数的一位、一个字节或一个字。存储单元必须能用指令给定的地址访问,并且一个地址码只能访问一个存储单元;反之,一个存储单元只能被一个地址访问。存储单元也指由地址确定的存储器的一个区域,可以是一个字块或字段的实际存储区。存储单元和位单元是不相同的,位单元是存储一位二进制数的信息单位,所以它的数值只能是1或0。在半导体存储器芯片中的一个位单元有时也称为一个存储单元,因为它是一个地址访问芯片时所涉及的信息单位。

存储器地址用以标识存储体和存储单元的编号或位置。一台存储器由一个或多个存储体组成,每个存储体中有多个存储单元,它们都有自己的地址编号,这个编号就是存储位置的地址。计算机最常用的信息存取方式是按地址存取,即CPU首先指定存储体编号和存储单元的位置号,再按照地址将信息存储到指定的存储单元里,或从指定的存储单元中取出信息。存储体采用三维阵列结构形式,每个存储单元的位置由笛卡儿坐标(即 x 、 y 和 z 三个方向坐标)定位。现今半导体存储器芯片已将地址译码器、驱动器和读写控制等外围电路与存储单元阵列集成在同一芯片上,芯片内地址译码器采用二维译码方法,它就是立方体的 x 和 y 方向的译码。另外设计选片译码器作为 z 方向坐标定位存储板和存储器芯片,这样的译码系统使相关联的信息线较少,

成本可降低。

半导体读写存储器有两种基本操作:读操作和写操作。在执行读操作时,由CPU发出读命令,并在地址总线上送出地址码,经地址缓冲后的低位地址直接加到存储器芯片的地址端,而高位地址经片选译码器进行译码生成片选信号,经过取数时间之后将被选择存储单元各位的数据同时出现在数据总线上,被CPU所接收。在进行写操作时,由CPU发出写命令,同时通过地址总线将地址码和数据送给存储器,将数据写入所指定的存储单元里。所以,存储器的地址总线是单向地址总线,数据总线是双向数据总线。

衡量一个存储器的性能主要有以下几项技术指标:

(1) **存储容量** 存储器所容纳二进制信息的总量称为存储容量。通常用字数 \times 位数或用字节数表示,前者常以Mb为单位,后者常以KB、MB和GB为单位。

(2) **存储周期** 连续两次访问存储器时所允许的最短时间间隔,也就是对存储器执行一次完整的存取数据操作所需最小时间。它是表征存储器工作速度特性的重要指标。依据存储器的工作状态不同,存储周期可以是读周期、写周期和读修改写周期。

(3) **取数时间**(又称读出时间) 从读命令开始到所读出的信息稳定在数据锁存器的输出端为止的时间间隔。它表示CPU发出读命令之后需要等待多长时间才能从存储器得到读回的数据,对CPU的设计和加快信息处理有重要意义。

(4) **存储器可靠性** 要求存储器在规定条件下能够长期稳定、可靠地运行,通常以**平均故障间隔时间(MTBF)**来衡量存储器的可靠性。MTBF越长,表明存储器的可靠性越高。提高可靠性的主要措施有对元器件进行严格老化和筛选,采用自动校正错误的编码技术、冗余技术等。

随机存取存储器(RAM)分为静态RAM(SRAM)和动态RAM(DRAM)。对于要求存储容量不大,存取时间较快的存储器采用SRAM芯片,存储器的系统设计较简单,但每信息位价格较高。DRAM集成度高,价格便宜,适用于较大容量存储器,其缺点是需要定期刷新,时序和控制线路均较复杂。在微型计算机中主要使用DRAM。

半导体读写存储器除了用于构成计算机主存储器外,还用来构成高速缓冲存储器,以及性能上与磁盘存储器相类似的半导体盘(固态硬盘)。

参考文献

郑筠编著. MOS 存储系统及技术. 北京: 科学出版社, 1990
(郭志先)

banjiegouhua shuju

半结构化数据 (semistructured data) 介于完全结构化数据 (如关系数据库与面向对象数据库中的数据) 和完全无结构数据 (如音像数据与自然语言文本数据) 之间的数据。

和关系数据或对象数据不同, 半结构化数据是自描述的数据, 它的模式不是统一的、预先给定的, 而往往是不规则且经常变动的, 并且它的数据和模式是混合存放。这种特点使得半结构化数据有很大的灵活性, 能够满足不同应用、不同企业之间交换信息的需要, 但同时也给半结构化数据存储、查询处理带来了很大困难。

目前半结构化数据模型可分为两类: 基于逻辑的描述形式和基于图的描述形式。其中, 基于图的模型方式是最常用, 最有效的。

最有影响的基于图的模型是对象交换模型 (OEM), 它最初在斯坦福大学的异构数据集成系统 TSIMMIS 中引入。OEM 是一个简单的、自描述的、嵌套的对象模型。从结构上看, OEM 是一个带标记的有向图, 结点表示对象, 每个对象具有惟一的对象标识, 边表示了对象间的关系。对象分为原子对象和复合对象两种, 原子对象包含一个原子类型的值, 如整数、实数、字符串等。复合对象的值是有序对 < 标记, 子对象标识 > 的集合, 每一个标记是对象与子对象之间关系的描述。每个 OEM 图都有一个根结点, 从根结点至任意结点都应是可达的。

图 1 表示一个简单的对象交换模型。它描述了

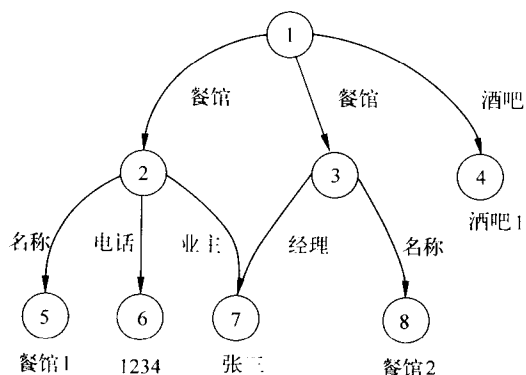


图 1 一个对象交换模型

某个餐饮业的结构。图中每个对象都有一个整型的对象标识, 根对象包括了三个子对象, 两个餐馆和一个酒吧。每个餐馆都是复合对象, 而酒吧是一个原子对象, 字符串“酒吧 1”是该对象的值。

目前, 半结构化数据研究集中在半结构化数据模式的提取; 半结构查询的处理和优化; 半结构化数据集成; 半结构化数据的存储、索引等。由于万维网上半结构化信息的不断丰富, 半结构化数据的研究是一个颇有前途的领域。

参考文献

1. Buneman P. Semistructured data. Arizona: ACM Press, 1997
2. Abiteboul S. Querying Semi-Structured data. Greece: Springer-Verlag, 1997 (高军 杨冬青)

bangding

绑定 (binding) 一个对象 (或事物) 与其某种属性建立某种联系的过程。如: 一个变量与其类型或值建立联系, 一个进程与一个处理器建立联系等。这种联系的建立, 实际上就是建立了某种“约束”。

绑定主要用于程序设计语言、数据库等方面。

在程序设计语言中, 它指把数据名转换为机器地址的过程, 把类型或值赋予变量或参数的过程等。

在数据库系统中, 则指把数据的一种视图转换为另一种视图的过程。在应用程序可以使用数据之前, 先要把应用程序的局部逻辑视图 (子模式) 绑定到数据库的全局逻辑视图 (模式), 尔后再绑定到物理视图 (存储模式)。绑定可以早在编译时进行, 也可以迟至取数据时进行。一般而言, 及早进行绑定可提高运行效率, 但适应修改的灵活性就较差; 延迟绑定则可取得相反的效果。程序设计语言的设计必须在灵活性和效率之间求得平衡, 绑定时间的控制正是达到这种平衡的一个重要手段。

绑定还分静态绑定和动态绑定, 静态绑定只需检查程序正文就可以判定绑定出现与给定的应用出现是否对应。至于动态绑定, 绑定出现与给定的应用出现是否相对应要取决于程序的动态控制流, LISP 和 Smalltalk 等语言有动态绑定, 即有动态类型, 大多数程序设计语言 (包括 FORTRAN, ALGOL, Ada, PASCAL 等) 均选择了静态绑定。

参考文献

岳东, 李南. 微型计算机高级程序设计语言的分
类与剖析. 北京: 海洋出版社, 1992 (程虎)

bijiben jisuanji

笔记本电脑 (notebook computer) 外形和笔记本相似的个人计算机。笔记本电脑的平面尺寸与 A4 复印纸的尺寸相当,其厚度在 3 cm 左右。打开计算机的顶盖,即露出操作键盘,在顶盖内部装有平板式液晶显示器,整台计算机的重量,连同机内电池在内,一般只有 3 kg 左右。笔记本电脑的基本配置包括中央处理器、存储器、软磁盘驱动器、硬磁盘驱动器、显示器、键盘、电源等。它和台式计算机在功能上没有什么差别,只是它的集成度更高,功耗更低,体积更小,重量更轻。而且,随着计算机制造业整体技术的不断提高和创新,笔记本电脑将更加微型化和轻型化。

笔记本电脑是便携式计算机的一种,便携式计算机分为膝上型、笔记本型和掌上型 3 类。1985 年膝上型计算机开始进入市场,它可以看成是台式计算机小型化以后的直接产物。它的外形像一个小包的公文包,功能齐全,接口丰富。用平板式显示器替代了台式机的阴极射线管显示器,用电池(一般为可充电电池)供电,电池的使用寿命不到 1 h。连同电池在内,整台机器的重量为 5 ~ 10 kg。膝上型计算机经常在流动环境中使用,没有桌子时,可坐下来放在膝上使用,所以称为膝上型计算机。1988 年,日本 NEC 公司推出了一种设计独特的、称为 Ultraline 的膝上型计算机,其大小和一个 A4 纸张尺寸的笔记本相当,具备 IBM PC AT 机的功能,重量不到 5 lb (1 lb ≈ 0.4536 kg),宣告了笔记本电脑的诞生。

笔记本电脑的技术特点

笔记本电脑的主要技术特点体现在以下几个方面:

(1) 中央处理器 笔记本电脑的中央处理器大都采用 Intel 80x86 系列的微处理器,也有少量采用 Motorola 公司生产的 68000 系列,或 Power PC 系列的微处理器,或其他 RISC 处理器。以 Intel 系列的微处理器为例,Intel 公司在便携式计算机用的中央处理器中,增加了能源管理技术。这种技术以 Intel 公司特有的系统管理模式(SMM)为基础,在中央处理器中增设了不可屏蔽的系统管理中断(SMI)以及与之相对应的复苏(resume)指令和相应的系统管理程序。这种扩充了能源管理功能的处理器,称为 80x86 SL 处理器。当具有 SL 结构的微处理器运行在 SMM 模式时,可以提供透明于操作系统和应用程序的电源管理功能。可实现对中央处理器、主存、

外围设备工作状态的控制,使之处于全工作状态、半工作状态或休眠状态(即闲置状态)。这些状态的控制均独立于操作系统和应用程序,状态间的切换可瞬间完成,达到完美的节电效果,又不给系统程序员和应用程序员带来任何麻烦。这种技术一般可使笔记本电脑的电池寿命提高 1 倍。

(2) 显示屏 笔记本电脑一般采用平板式液晶显示器,它是笔记本电脑中成本最高的部件,约占整机成本的 30% ~ 40%。1993 年以前,一般采用单色显示器,少数采用无源单扫描矩阵彩色显示器,色彩效果不理想,对比度和亮度较小,刷新时间长,画面移动时会留下踪迹。1993 年以后,开始采用双扫描技术,把屏幕分成两个可以同时刷新的部分,彩色较鲜艳,但是,视角较小。后来,又出现了有源矩阵彩显(用薄膜晶体管 TFT 显示),这种显示屏具有图像清晰、色彩鲜明、亮度高、无阴影、视角较大等优点。但是,它的成本高,价格昂贵,耗电和重量也较大。此外,笔记本电脑的屏幕分辨率一般为 640 × 480,明显低于台式计算机,因此,一般笔记本电脑都备有连接台式计算机显示屏的接口。

(3) 存储器 笔记本电脑的存储器分主存和外存。主存可用 DRAM 或 SRAM,前者价格低,集成度高,后者省电,更适合于笔记本电脑。外存采用 3.5 英寸的软磁盘驱动器。硬磁盘采用 2.5 英寸或 1.8 英寸的小型磁盘,其体积小、功耗低。后来半导体盘(又称快擦写存储器)已作为硬磁盘的替代物进入便携机市场。它用 Flash EEPROM(参见快可擦编程只读存储器芯片)作存储介质,整片擦除只需一秒钟,有的半导体盘每秒钟可写入 100 kB 的数据,平均存取时间小于 1.5 ms,擦写次数可达数万次到 100 万次。半导体盘在读写速度、存取时间、功耗、体积、重量、耐冲击力等方面比硬磁盘更适用于笔记本电脑。但是,在存储容量和价格方面仍不如硬磁盘。

(4) 电源 一般笔记本电脑既可用电池供电,也可用交流市电供电。电池一般用可充电电池,早期采用镍镉电池,每次充电前必须先放电,使用不方便。后来使用镍氢电池,它无记忆效应,使用较方便,而且单位重量的电量较大。其后出现锂离子电池,它的使用时间长,但价格比镍氢电池高。在同样重量情况下,上述 3 种电池的使用时间比为 1:1.2:1.9。锂离子电池 1 次使用时间为 4 ~ 6 h。为了延长电池的使用时间,笔记本电脑普遍采用智能电源管理技术,如在不需要的情况下降低中央处

理机的运行速度,降低总线工作频率,减弱屏幕亮度,暂停闲置的硬盘驱动器,关闭未使用的通信端口的电源等。

(5) 接口 为了获得台式计算机的扩展卡所提供的灵活扩展能力,笔记本计算机普遍采用个人计算机存储卡国际协会 PCMCIA 所制定的标准。该标准定义了数据存储卡和外围扩展卡的类型,使得笔记本计算机能够灵活地插入标准的 PCMCIA 设备进行工作。这种 PCMCIA 设备也叫 PCMCIA 卡,简称 PC 卡。PC 卡分 I, II, III 型,大小类似于信用卡,长度均为 85.6 mm,宽度均为 54.0 mm,厚度分别为 3.3 mm, 5 mm 和 10.5 mm。为了与 I 型卡的端口兼容,II 和 III 型卡的边缘部分仍为 3.3 mm。PC 卡分为通用卡和特殊功能卡两类。前者有存储卡(静态随机存储器、只读存储器、可擦[可]编程只读存储器、快[可]擦编程只读存储器等)、传真及调制解调器卡、网络适配器、语音卡、手写体识别卡、多功能卡、硬盘驱动器;后者有数据加密卡、全球定位系统接收卡等。PC 卡大大拓展了笔记本计算机的应用范围。

(6) 其他 笔记本计算机的鼠标和键盘也颇具特色。鼠标最初与台式计算机的一样,为带线外接型。后来改为可拆卸的轨迹球,使用时插入机器底座的右侧,接着又演变成固定在键盘上的某个位置。其后,IBM 公司推出的压敏式指挥杆,采用压敏技术,顶端直径仅 7 mm,使用时只要轻压杆顶即可自如地改变光标位置,效果比跟踪球好。笔记本计算机的面积接近于台式计算机键盘的主体部分,因此,笔记本计算机的键盘布局与台式计算机的键盘的主要部分相似,辅助键的布局,各厂商有各自的设计。由于台式机的键盘按人体工程学原理设计,键盘分布合理,长期使用不易疲劳,这限制了笔记本计算机向亚笔记本计算机的发展。在键盘设计中,IBM 公司发明了伸展式键盘,打开机器后,键盘沿对角线分成两部分展开,最后变成比笔记本计算机宽度更长的键盘,以便有效地布置键盘上的辅助键,也可以为亚笔记本计算机装上与笔记本计算机类似的键盘。

笔记本计算机的发展趋势

笔记本计算机体积小、重量轻、耗电低,只占很小的桌面空间,便于随身携带或在室内搬动,为人们提供了可移动的工作环境,它的一体化机芯、低功耗器件以及精密组装技术,使得它的可靠性、可用性和可维护性优于台式计算机。它的显示屏消除了台式

机阴极射线管屏幕对人体的辐射。标准接口和 PC 卡的日趋成熟、齐全,使笔记本计算机的扩充性能不亚于台式计算机。这一系列的明显优点促使笔记本计算机的市场发展迅速,占有了越来越多的传统台式微型计算机的市场份额。但笔记本计算机的价格较贵,真彩色显示屏占了整机成本的 40 % 左右。因此,增大彩色液晶显示屏面积、提高分辨率和降低成本是需要解决的问题。

笔记本计算机将继续沿着更轻、更薄,功能、性能和电池寿命均有所提高的方向发展。在功能扩展方面,多媒体化和网络通信化是笔记本计算机的发展趋势。家用电器(例如激光音响、便携电话、电视、摄像机和录像机等)的功能将融入笔记本计算机中。除了键盘输入外,将增加语音输入和笔输入等人机接口功能。此外,更多的应用,如电子记事簿、全球定位系统的接收机以及个人通信,也将在笔记本计算机中日趋完善。

参考文献

Ralston A, Reilly E D Edited. Encyclopedia of Computer Science. 3rd Ed. New York: IEEE Press, 1993

(韩承德)

bianjie wangguan xieyi

边界网关协议 (Border Gateway Protocol, BGP) 一种在 Internet 中用于自治系统间的外部路由协议。Internet 中最早采用的外部路由协议是外部网关协议(EGP),它发表在征求意见稿 RFC 827 中。由于 EGP 有不少缺点,1989 年在 RFC 1105 中又发表了 BGP 来取代它,后来又在 RFC 1163、RFC 1267 和 RFC 1268 等文献中做了修改与补充。1994 年在 RFC 1654 中发布了最新版本的 BGP 4,在 RFC 1771 中有有关 BGP 的极为详细的资料。

边界网关协议(BGP)与距离向量协议类似,它与采用距离向量算法的距离向量协议(如 RIP 协议)的不同之处在于邻点间交换的不是简单的距离向量,而是到达目标的完整的路由,因而不存在距离向量协议固有的收敛慢的缺点(参见路由选择)。BGP 在选择路由时还要考虑策略。策略可以由人工来配置到 BGP 路由器中。策略通常是综合考虑政治、安全和经济等诸方面的因素决定的。比如说从加拿大一处到另一处的报文路由不能出国境;始于或止于 IBM 的通信不能经过 Microsoft;由 A 发出的报文尽可能经过 B,因为 B 是合作伙伴且有互免收费的双边协议等。这些策略都可以通过打分或加

权而反映到最后的**路由选择**过程中去。

BCP 4 中共使用了四种报文: Open 报文, 用来探测邻点; keepalive 报文, 用来回答 Open 报文, 并周期性地确认邻接关系; Update 报文, 用来发送某一路由信息或撤销某些路由; Notification 报文, 用来发送检测到的差错。通过这些报文复杂的交换过程, 使得每个路由结点都能获得网络状况的最新信息, 并依据策略来选择新的路由。

参考文献

Perlman R. Interconnections. Second Edition. MA: Addison Wesley, 2000 (高传善)

bianji chengxu

编辑程序 (editor) 对文件内容进行增加、删除及修改等操作的程序。它也可以用于创建、清除和复制文件本身。被编辑文件的基本元素可以是字符、文字、表格、图形及图像等。编辑对象是字符、文字和表格的编辑程序称为**正文编辑程序**, 它常用于源程序的编辑。除字符、文字外, 还可编辑图形的编辑程序称为**图形编辑程序**。以图像为主要编辑对象的编辑程序称为**图像编辑程序**。

编辑程序的用户界面包括输入、输出设备和交互语言。用户通过输入设备选择要编辑的文件、输入编辑命令和编辑元素; 通过输出设备获取被编辑的元素和编辑的结果; 交互语言定义规则和设施以使用户与编辑程序进行交互。编辑过程即是在用户编辑命令控制下对文件进行创建、删除、修改、移动、复制及显示打印的过程。

编辑程序的逻辑结构可分为输入、命令解释和输出三个部件。输入部件接受输入设备送入的字符、保存命令行及对功能信号进行译码; 命令解释部件分析输入部件送来的信息, 识别、解释相应的编辑命令、执行编辑动作; 输出部件将编辑的结果写到输出设备上。

编辑程序在 20 世纪 60 年代即已出现, 几十年来, 随着计算机应用领域的拓广, 编辑程序的功能不断增强。以正文编辑程序为例, 就视域而言, 从行编辑扩展到全屏幕编辑, 又扩展到多窗口编辑; 就编辑的正文结构而言, 已从“无结构”扩展到“有结构”, 如语法制导编辑; 就处理能力而言, 已不仅限于对文件内容的增、删和改, 还提供诸如拼写检查、预浏览等创建和打印高质量文档的能力, 如字处理程序。随着编辑程序功能的不断扩大, 可望应用手写体识别、语音识别和图像识别等多媒体技术以使得人机

界面变得尽可能丰富、自然和直觉。

参考文献

Beck L. L. System Software: An Introduction to Systems Programming. 2nd ed. Addison-Wesley, 1990

(张素琴)

bianma gongju

编码工具 (coding tool) 软件开发过程中用于编码活动的工具。它辅助程序员采用某种程序语言编制源程序, 并对源程序进行翻译, 最终转换成可执行的代码。通常, 编码工具与编码所使用的语言密切相关。

编码工具主要有编辑程序、汇编程序、编译程序和生成程序等。

编辑程序用以对文件内容进行增加、删除及修改等操作, 这类编辑程序主要用于辅助编制源程序。除通常的以字符为单位进行编辑的正文编辑程序外, 还有面向程序语言语法单位的语法制导编辑程序和混合编辑程序。语法制导编辑程序可根据程序语言的语法规则提供编辑时的语法指导和检查; 混合编辑程序兼有正文编辑和语法制导编辑两种方式。

汇编程序用以将汇编语言书写的程序翻译成等价的机器语言程序。如果汇编程序所生成的机器指令代码是另一种计算机的机器指令, 这类汇编程序便称为交叉汇编程序。

编译程序用以将高级语言书写的程序翻译成等价的低级语言程序。

生成程序通常根据与领域有关的甚高级语言或某种专用语言描述的用户需求, 自动生成高级程序语言或汇编语言描述的程序。例如, 词法分析程序 LEX 根据正规表达式表示的词法规则, 自动生成高级程序语言词法分析程序的代码段, 来实现能识别所说明的正规表达式的有限状态自动机。

此外, 还有一些编码辅助工具, 例如以语法分析为基础的美观打印程序, 它能使输出的源程序层次分明。

(钱乐秋 洪梅)

bianyi chengxu

编译程序 (compiler) 将高级语言书写的程序翻译成等价的**机器语言**程序或汇编语言程序的处理系统。

编译程序以高级语言书写的程序作为输入, 称

之为源程序;而以机器语言或汇编语言表示的程序作为输出,称之为目标程序;其最终任务是产生一个可在具体计算机上执行的目标程序。执行目标程序将会按照用户在源程序中所规定的意图,加工初始数据,算出所需的结果,完成所希望的加工任务。源程序中的每个语句与目标程序中的指令通常是一多对应关系,所以编译程序的实现算法较为复杂,但它可以产生高效运行的目标程序,因此编译程序更适合于翻译那些规模较大、结构较复杂、运行时间较长的大型应用程序。

编译程序必须分析源程序,然后综合成目标程序。为此,编译程序要在分析阶段建立符号表、常数表和中间语言程序等数据结构,以便在分析和综合时引用和加工(图1)。源程序的分析是经过词法分析、语法分析和语义分析三个步骤完成的,目的是检查源程序的语法和语义的正确性,并把源程序分解成一系列的基本组成成分。目标程序的综合通常包括存储分配、代码优化、代码生成等几个步骤,目的是为源程序的常数、变量、数组等数据结构分配存储空间,重新组织分析阶段产生的基本组成成分,将其综合成高效运行的可执行目标程序。

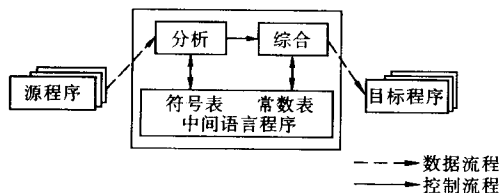


图1 编译程序工作过程示意图

编译程序在逻辑上由分析和综合两大部分组成,并可进一步细分为词法分析、语法分析、语义分析、存储分配、代码优化和代码生成6个相继的逻辑步骤。具体设计和实现编译程序时,通常是按照从头到尾扫视源程序(或其等价的中间语言程序)的遍数来规划编译程序的结构,安排相关逻辑步骤的工作。每一遍可以按顺序执行方式或并行调用方式,完成一个或相连几个逻辑步骤的工作。例如,可以把词法分析作为第一遍;语法分析和语义分析作为第二遍;存储分配和代码优化作为第三遍;代码生成作为第四遍。反之,为了适应较小的内存空间或提高目标程序质量,也可以把一个逻辑步骤的工作分为几遍去完成。例如,代码优化可划分为代码优化准备和实际代码优化两遍来完成。编译程序采用多少遍的编译结构,应根据机器的规模、程序语言的

繁简、编译程序的功能、目标程序的质量、设计人员的多少等具体情况而定。一遍编译程序是一种极端的情况,整个编译程序同时驻留在内存,彼此之间采用调用转接方式连接在一起工作(图2)。当语法分析程序需要新符号时,它就调用词法分析程序;当它识别出某一语法结构时,它就调用语义分析程序。语义分析程序对识别出的结构进行语义检查,并调用“存储分配”和“代码生成”程序生成相应的目标语言的指令序列。

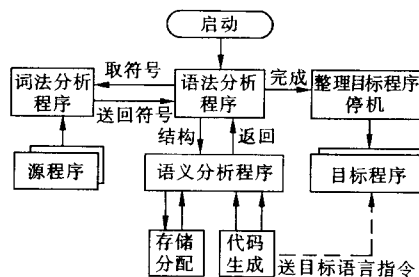


图2 一遍编译程序

随着高级语言在形式化、结构化、智能化和可视化等方面的发展,作为实现相应语言功能的编译程序,也随之向自动程序设计和可视化程序设计的发展方向,为用户提供更加理想的程序设计工具。

参考文献

1. Gries D. Compiler Construction for Digital Computer. New York: Wiley, 1971
2. Aho A V. Compilers Principles, Techniques and Tools. New York: Wesley, 1986 (曹东启)

bianyi chengxu de bianyi chengxu

编译程序的编译程序 (compiler-compiler)

产生编译程序的编译程序。它接受用某一适当的表示体系描述的某一语言类中任一语言A的词法规则、语法规则、语义规则和(或)代码生成规则,并从这些描述产生出用目标语言B写的关于语言A的编译程序的全部或部分。编译程序的编译程序又称为编译程序的生成程序。

通常,编译程序分成词法分析程序、语法分析程序、语义分析程序、代码生成程序等若干部件。这些部件可以用一个总的编译程序的编译程序的不同部分来生成,也可以分别用不同的专用生成程序来生成。这些专用生成程序包括词法分析程序的生成程序,语法分析程序的生成程序和代码生成程序的生

成程序等。

词法分析程序的生成程序接受以正则文法或其他类似文法描述的单词,构造一个有限状态自动机,由此生成一个词法分析程序。

语法分析程序的生成程序接受以上下文无关文法的形式描述的源语言的语法,生成一个语法分析程序。各种语法分析程序的生成程序随实现语言、语法分析算法的原理以及源程序中语法错误处理能力的不同而异。如采用 LR 的语法分析程序的生成程序 YACC 和采用递归下降法的语法分析程序的生成程序 LLgen。

语义分析程序、中间代码生成程序和目标代码生成程序的生成程序的设计与实现与形式化的语义描述紧密相关。语义描述形式化技术相当困难,目前大多数生成程序中语义描述还是采用非形式化,它们的基本思想是为源语言的上下文无关文法的语法符号或产生式配以翻译子程序(语义动作或语义子程序)。

现有不少性能很好的编译程序的编译程序,如词法分析程序的生成程序 LEX,语法分析程序的生成程序 YACC 和 LLgen,它们都显著提高了编译程序的开发效率。

参考文献

1. 陈火旺,钱家骅,孙永强. 程序设计语言编译原理. 第二版. 北京:国防工业出版社,1984
2. Aho A V, Sethi R, Ullman J D. Compilers Principles, Techniques and Tools. Addison-Wesley, 1986 (徐永森)

bianliang

变量(variable) 具有“值”这一属性、且可视需要对其值进行更新的计算对象。变量可用作诸如世界人口、某人年龄、当前天气等具有状态的现实对象的模型。

有两种变量,一种可在程序内部创建并使用,且可通过赋值更新其值,生存期较短;另一种如文件、数据库等,生存期较长,其存在和特定程序无关,其更新是有选择的。

程序设计语言中的变量和数学变量不同,前者可以更新其值,后者代表一个固定但未知的值,该值并不能改变。

随着语言不同,有的变量无类型,有的变量有类型。变量的具体类型规定了它的取值范围,如果由于某种原因被赋予的值超出了这一范围,就能被发

现,所以有类型的变量较为安全。(参见数据类型)。

参考文献

- Watt D A. Programming Language Concepts and Paradigms. Prentice Hall, 1990 (徐家福)

bianxingti donghua

变形体动画(soft object animation) 采用变形技术作为运动控制的动画技术。变形可以是二维的或三维的,可以是几何的或图像的。传统动画的一个显著特点是赋予每个角色以个性,并以形状变形来渲染某些夸张的效果。虽然传统动画的有些夸张效果用三维动画还很难实现,但**计算机动画**的研究者们已在形状变化方面做了不少出色的工作,并在电视、电影、广告和音乐电视(MTV)中得到了广泛的应用。大部分变形方法与物体的表示有密切关系,如通过移动物体的顶点或控制顶点来对物体进行变形。为了使变形方法能很好地结合到造型和动画系统中,人们提出了许多与物体表示无关的变形方法。

变形体动画可分为形态变换和空间变形两大类。形态变换是指将一给定的源数字图像或几何对象 S 光滑连续地变换到目标数字图像或几何对象 T。在这种光滑过渡中,中间帧应既具有 S 的特征,也具有 T 的特征。S 和 T 的拓扑可以相同也可以不同。形态变换通常需要动画师指定源处理对象和目标处理对象之间特征的对应关系,当然,这种对应关系也可以由系统自动求得。空间变形指将单个几何对象的形状做某种扭曲、变形,使它变换到动画师所需的形状。在这种变化中,几何对象的拓扑关系保持不变。空间变形更具有某种随意性,所以也常称为自由变形。空间变形包括与物体有关的变形和与物体无关的变形。

参考文献

1. 唐荣锡,汪嘉业,彭群生等. 计算机图形学教程. 北京:科学出版社,2000
2. 鲍虎军,金小刚,彭群生等. 计算机动画的算法基础. 杭州:浙江大学出版社,2000 (金小刚)

biaozhundai

标准带(standard tape) 用来调整磁带机的互换性和检测磁带性能用的工具磁带。又称参考带。不同带宽、不同记录格式、不同用途的磁带机各

有相应的标准带。对于常用的 12.7 mm (0.5 in) 的数字记录用磁带,有 3 种标准带:

(1) **标准幅度带** 带上没有任何信号,但磁带的物理性能、机械特性和电磁特性都符合国际标准化组织的有关标准。因此可用来检测磁带机的读写性能,也可用来作为标准检测磁带产品的性能。

(2) **标准扭斜带** 带面上所有磁道都有连续的 1,每一行的信息都精确地排成一列,并与磁带的基准边相垂直。它用来调整磁头的方位角,检测磁头各道的位置偏差和磁带机读写电路的时间延迟。

(3) **标准速度带** 带上记录的也是连续的 1,但要求各行间距离精确。它用来检测磁带机走带速度的平均误差和瞬间误差,以保证互换性。

一级标准带由国际标准化组织指定专门的单位制造和保管,所指定的单位还负责提供二级标准带。二级标准带的制造方法是挑选性能良好的磁带,录入信息后进行测试,将得到的数据与一级标准带有关参数相比较,求得各参数的偏差值。这种参数偏差值已知的磁带就作为二级标准带使用。实际生产线测试用的是三级标准带,它的参数偏差值是其测试所得数据与二级标准带有关参数相比较得到的。由于磁带在工作过程中有磨损,标准带在使用一定次数后就报废,不再使用。

参考文献

杨萍,邢班龙,林兼. 关于信息交换用磁带记录方面的标准. 电子计算机动态, 1974(3) (林兼)

biaozhun tongyong zhibiao yuyan

标准通用置标语言 (standard generalized markup language, SGML) 一种用于定义置标语言的语言。标准通用置标语言 (SGML) 主要关注文档的嵌套结构及其所涉及正文的标记方法。一般而言,一个文档具有结构、内容和样式三个方面,而 SGML 提供了将这三个方面分开来予以描述的标记法,同时也提供了将这三个方面相互联系起来的机制。

结构 SGML 可以用来定义嵌套结构的文档类型。一个文档类型定义 (DTD) 定义了一个文档类型,规定了能使用的置标词汇和使用这些词汇的规则。例如,对标题、段落、表、表格、图形和音频等,给出它们的段落格式,这些条目、段落的出现顺序,以及相关的置标标记。

内容 内容是文档中的正文,是文档所含信息本身。为了标识有关内容在对应 DTD 结构中所起

的作用,需要在有关正文内容开始和结尾处,插入相应的标记。例如,“<par>”指明一个段落的开始,而“</par>”指明一个段落的结束。

样式 SGML 文档的印刷或联机出版通常需要用样式表文件来指定形象化格式。但 SGML 语言本身并没有为样式设置任何标准。

SGML 具有 3 个显著特性: ①描述性 注重描述文档结构及其属性,而不是指定作用在文档之上的操作。②定义类型 对文档提供类型定义。可以利用 DTD 文档类型,对具体的文档实例进行有效性检查。③数据独立性 为克服不同字符集的差异性问题,提供了一种替代字符串的通用机制。以上 3 个特性使得 SGML 适合跨多机平台的应用与异构环境下的应用。

超媒体-时基结构化语言 (HyTime) 是 SGML 的一个重要应用。它在超文本和多媒体应用中,为处理和交换静态和动态信息提供了基础。它提供了标准,为指定文档内部各信息对象之间的超链接,以及安排多媒体信息的时序衔接和空间衔接等表示机制。HyTime 可以用来表示 SGML 文档内部的或者 SGML 文档之间的超链接。

参考文献

1. ISO 8879-1986. Information processing—Text and office systems—Standard Generalized Markup Language (SGML). [Geneva]: International Organization for Standardization, 1986
2. Goldfarb, Charles. The SGML Handbook. Oxford University Press, (1990) ISBN 0-19-863737-9
3. ISO/IEC 10179-1996 (E). Information technology—Processing languages—Document Style Semantics and Specification Language (DSSSL). [Geneva]: International Organization for Standardization, 1996
4. ISO/IEC 10744-1992 (E). Information technology—Hypermedia / Time-based Structuring Language (HyTime). [Geneva]: International Organization for Standardization, 1992 (瞿裕忠)

biao chuli yuyan

表处理语言 (list processing language) 一种用于描述表及其处理的程序设计语言。表是一种特殊数据集,它是由若干个(包括零个)元素组成的序列。此种数据结构灵活、方便、通用。

表处理的基本操作有:将新元素插入表的指定位置中;访问表中任一元素;删除表中指定位置的元

素;将两个表链接成一个新表;将一个表拆成两个表;判断指定表是否空。

第一个表处理语言是1958年由A. Newell等研制的IPL-V。最典型的表处理语言是1960年由J. McCarthy及其研究小组设计实现的LISP。其他表处理语言还有Slip, Pop-2等。20世纪70年代末以来出现了几十种LISP版本,并建立了国际标准Common LISP。在面向对象技术热潮推动下,1988年美国学者将当时若干有代表性的LISP合并为CLOS,使之成为AI领域重要的语言。

参考文献

1. Newell A (Ed.). Information Processing Language-V Manual. Englewood Cliffs: Prentice-Hall, 1961
2. Winston P H, Horn B. LISP. 3rd Ed. Reading: Addison-Wesley, 1989
3. Steele G L. Common LISP. 2nd Ed. Bedford: Digital Equipment Coration, 1990 (郭浩志)

biaodashi

表达式 (expression) 高级语言中用来指明求值对象与规则的基本语法成分。

表达式可以是简单的,也可以是复杂的,但一般都涉及到参与计算的运算对象(或称为操作数),进行计算的运算符,也可以有指明求值次序的圆括号。

表达式的运算对象可以是无正负号常量,变量,函数命名符,或由圆括号括起来的另一层表达式。变量可以由单个标识符标记的整体变量,也可以是构造类型的成分变量(如数组的下标变量,记录的域变量等)。

运算符用于对运算对象的求值。若按参与运算的对象类型来分类,则可分为算术运算符、关系运算符、逻辑运算符及集合运算符,而从运算涉及到的运算对象个数来看,也可分为单目运算符和双目运算符,双目运算符又可分为乘除运算符、加减运算符、关系运算符及逻辑运算符等。同一层表达式的各种运算符,一般按数学上的先乘除后加减的原则来定义它们的优先级。例如,Ada语言中对运算符定义了6个优先级:乘幂(* *),绝对值(ABS),非(NOT)等单目运算符优先级最高,AND,OR,XOR等逻辑运算符优先级最低。同一层表达式的计算一般从左向右进行,但优先级高的运算符先做。

表达式可分为同构型表达式和混合型表达式。同构型表达式要求其所有成分都属于同一类型,例

如,ALGOL 60中定义三类表达式,即算术表达式、布尔表达式及命名表达式,均属于同构型表达式,但目前多数语言的表达式则为混合型表达式,例如,Modula-2, Ada等语言,一个表达式中也许有不同类型的运算对象,它们通过类型转换来解决运算对象类型不一致问题。

有类型语言要求参与计算的运算对象、运算符是有类型的,因而其表达式也是有类型的。运算符的类型可以从运算符的符号不同来区分,例如“÷”为整数类型运算符,“/”为实数类型运算符,但当前不少程序语言(如Ada, C++等)中,同一个运算符,其类型根据其运算对象类型的不同可以作不同的解释。例如,Ada语言中的“=”运算符,其类型根据其运算对象类型,可解释为整数相等比较,也可解释为实数相等比较,甚至可以是用户定义的某一类型的相等比较,这种现象称为运算符的一名多用。

参考文献

- Ralston A, Reilly E D (Editors). Encyclopedia of Computer Science. third edition. New York: Van Nostrand Reinhold, 1992 (陈涵生)

biaomian anzhuang jishu

表面安装技术 (surface mounting technology, SMT) 用波峰焊或再流焊技术,将无引线或短引线的片状元器件(亦称表面安装元器件)焊到印制电路板或其他基板上的一种工艺技术。表面安装是在高速自动贴装机上进行的,其过程包括贴装和焊接两步。表面安装与通孔插装不同,表面安装时的元器件面与焊接面均在印制电路板或基板的同一面上。表面安装具有组装密度高、印制电路板或基板占用面积小、电气性能好、抗干扰能力强、可靠性高、成本低及易实现自动化生产等优点。它涉及材料(基板材料与工艺材料)技术、组装技术(贴装、焊接及清洗等)、设计技术、测试技术、标准化、可靠性等多种交叉学科技术。

1966年,美国RCA公司研制成功片式薄膜电阻,并用于微膜组件上。20世纪70年代日本将其应用于民用电子产品。80年代中期开始,我国电子行业开始引进、消化、吸收表面安装技术。进入90年代以后,表面安装技术作为微组装技术的一项关键技术,正处于蓬勃发展的新阶段。

表面安装技术包括表面安装元器件、表面安装工艺和表面安装设备三方面。表面安装技术的发展主要取决于表面安装元器件。

表面安装元器件可分为无源元器件、有源元器件、机电元器件和集成电路等几类。从外形上可分为矩形片式(扁平型)元器件和电极面焊型(圆柱型)元器件两大类。不论哪一类表面安装元器件,它们都应符合下列基本要求:元器件的尺寸、形状应符合标准化和自动化安装的要求;有足够的精度和一定的机械强度;具有良好的电气性能并耐有机溶剂洗涤等。

表面安装工艺有4种:①在印制电路板的一面全部贴装表面安装元器件的单面安装;②在印制电路板的两面全部贴装表面安装元器件的双面安装;③在单面安装印制电路板上又插装有引线元器件的单面混合安装;④在印制电路板的一面贴装表面安装元器件,另一面贴装表面安装元器件和插装有引线元器件的双面混合安装。表面安装元器件的贴装方法有顺序式、同时式、流水线式和顺序同时式4种。顺序式是将印制电路板装在 $x-y$ 工作台上,把表面安装元器件逐个顺次贴装。同时式是用盒式包装或料斗进料,通过模板成批同时贴装。流水线式是使印制电路板沿传送轨道,通过各个贴装头,将表面安装元器件依次贴装到相应的位置上。表面安装的焊接技术有再流焊和波峰焊两大类。再流焊的优点是热应力小,特别适用于半导体片式器件的焊接,其加热方法有红外加热、光束加热、汽相加热和激光加热等。波峰焊一般用于混合安装的有引线元器件的焊接,其缺点是对片式元器件热应力大,对有的片式元器件不适用。

表面安装的设备包括用来印刷焊膏或贴片胶的丝网印刷机、自动贴装机、再流焊机(或波峰焊机)、清洗机及检测设备等,其中自动贴装机是表面安装技术中的关键设备。

目前表面安装技术研究的重点在于高密度、高速度、细间距、高可靠组装设计、基板制造、焊膏及检测等工艺技术。

(赵悼爻)

biaoqing donghua

表情动画(facial expression animation) 通过计算机来模拟人的表情的动画技术。它实际上涉及脸部造型和脸部动画两个问题。这方面的研究内容包括脸部几何造型、由照片合成人脸、语音驱动表情动画、表情捕捉和重现、脸部纹理映射、脸部形态变换、脸部细节模拟、表情库和表情合成等。在脸部表情合成方面,一种简单的方法是数字化仪将人脸的各种表情输入到计算机中,然后用这些表情的线

性组合来产生新的脸部表情。该方法的缺点是缺乏灵活性,不能模拟表情的细微变化,并且与表情库有很大关系。1987年,Waters提出了一个基于Facial Action Coding System的脸部表情动画模拟方法。该方法由一个参数肌肉模型组成,人的脸用多边形网格来表示,并用肌肉向量来控制人脸的变形。它的特点在于可用一定数量的参数对模型的特征肌肉进行控制,并且不针对特定的脸部拓扑结构。Waters用该方法生成了高兴、恐惧、愤怒、厌恶、惊奇等逼真的表情,Waters的算法已成为模拟脸部表情动画的核心算法之一。

参考文献

1. 鲍虎军,金小刚,彭群生等. 计算机动画的算法基础. 杭州:浙江大学出版社,2000

2. Watt A, Watt M. Advanced Animation and Rendering Techniques: Theory and Practice. London: Addison-Wesley, 1992

(金小刚)

bingfa chengxu sheji

并发程序设计(concurrent programming)

一种程序设计方法,按这种方法设计时,一个程序由若干个可同时执行的程序模块组成,这些可同时执行的程序模块称进程。严格地说,这里指的是进程描述,非进程本身。进程由数据和有关的语句或命令序列组成。组成一个程序的多个进程可以多台处理机并行地执行,也可以在一台处理机上交叉地执行。采用并发程序设计可以提高计算机系统效率,缩短程序执行时间。对于多机处理系统,上述结论是明显的,对于单处理机系统也是正确的。

例如一个从磁盘读入数据、加工后打印输出的程序,采用并发程序设计后它由三个进程组成:读盘进程,加工进程和打印进程。三个进程可轮流地占用处理机执行程序。在打印进程启动打印机后,在打印机工作期间,打印进程就无需占用处理机而处理机可被加工进程占用来加工数据或被读盘进程占用来控制读入数据。这样就增加了外部设备和主机操作的并行度,从而提高了系统效率,缩短了程序的执行时间。

并发程序设计是在多道程序设计的基础上发展起来的。1968年,E. W. Dijkstra首先引入了并发程序设计的概念并研究了有关的同步和死锁问题。70年代人们开始将有关并发程序设计的概念引入程序设计语言中,出现了并发PASCAL,Modula和Ada等

并发程序设计语言。

采用并发程序设计时必须处理好进程同步和死锁。所谓进程同步是指有共享数据的若干个进程对它们的访问共享数据的一种约束。例如 P1 和 P2 分别是将数据送入缓冲 B 和从缓冲 B 读出数据的两个进程。如果 P1 和 P2 不按先送后读的次序运行,那么可能一个数据被 P2 两次读出或者一个数据未读出时就被新读入的数据冲掉了。为了防止产生这样的错误,操作系统为并发程序设计提供了同步机制。**PV 操作**就是具有代表性的同步机制。P 操作和 V 操作是操作系统提供的两条原语。所谓**原语**是指它的执行是不会有被打断的。执行 P 操作 P(S) 时信号量 S 之值减 1,若结果不为负则 P(S) 执行完毕,否则执行 P 操作的进程暂时停止执行等待释放。执行 V 操作 V(S) 时,信号量 S 之值加 1,若结果不大于 0 则释放一个因执行 P(S) 而等待的进程。有了 PV 操作后上面的例子所产生的问题就不难解决了。在上例中我们定义两个信号量 S1 和 S2,它们的初始值分别为 1 和 0。进程 P1 在送入数据前执行 P 操作 P(S1),在送入数据后执行 V 操作 V(S2)。进程 P2 在读取数据前先执行 P 操作 P(S2),在读出数据后执行 V 操作 V(S1)。当 P1 送入一数据后信号量 S1 之值变为 0,在该数据读出后 S1 之值才变为 1,因此在前面一数未读出前后面一数不会送入。这就保证了进程 P1 和 P2 的同步。

死锁给并发程序设计带来了另一个问题。死锁是指进程间因为竞争资源而产生的无休止的等待。例如有两个进程 P1 和 P2, P1 占有资源 A 而申请资源 B, P2 占有资源 B 而申请资源 A。显然它们两个都不能得到所需的资源而无休止地等待下去。解决死锁问题有两种途径,一是设计防止死锁的资源调度算法,另一途径是检测死锁使得当死锁发生时能及时发现并进行排除。

参考文献

Hansen P B 著. 并发程序的系统结构. 杨芙清等编译. 北京: 国防工业出版社, 1982 (孙钟秀)

bingfa chengxu sheji yuyan

并发程序设计语言 (concurrent programming language) 用于进行并发程序设计的程序设计语言。它的主要特征是引入了进程描述,因此,用它编写的程序指明了若干个可以同时执行的进程。此外这类语言还提供了进程同步、进程通信

和进程产生等功能。

进程同步功能主要靠系统提供的同步机制来实现。进程同步机制有 PV 操作、管程等。管程由一组可供所有进程访问的数据及有关的操作组成。只有引用管程操作是互斥地被引用,显然管程中提供为多个进程访问的数据是互斥地被各个进程访问的,这就是实现了同步控制。

进程首先的功能是指在程序执行的过程中一个进程可以产生出一个新的进程。与之相对应,系统也就提供撤销进程的功能。一般说,只有创建进程的进程才能撤销它所创建的进程。

迄今为止已有多种并发程序设计语言,如并发 PASCAL, 并发 C, Modula-2, Ada 和 Occam 等。

(孙钟秀)

bingfa kongzhi

并发控制 (concurrency control) 数据库管理系统 (DBMS) 中协调多个事务的并发执行的机制和过程。并发执行指多个事务在共同时间段内同时执行。

当多个事务并发执行时,即使各个事务分别能保证事务执行的正确性和数据库状态的一致性,但多个事务的交叉调度所形成的事务之间的相互影响可能导致不一致。常见的不一致性问题包括丢失更新、读脏数据、不一致的分析等。如果一个多事务并发调度与同一事务集合的某个串行调度执行过程等价,则称这个调度是可串行化的。可串行化的调度能保证事务执行的正确性和数据库状态的一致性。

DBMS 采用的保证事务调度可串行化的主要方法包括封锁方法、时间戳方法、基于有效性确认的方法。其中最常用的是封锁方法,其基本思想是,事务在对任何数据库元素进行读写操作之前都必须申请对该数据库元素的适当封锁,系统通过不同类型锁之间的冲突来控制不同事务对相同数据库元素的读写,若系统各事务对封锁的申请和释放遵守两阶段封锁协议,则能保证事务调度是可串行化的。

参考文献

1. Date C J 著. 数据库系统导论. 孟小峰, 王珊等译. 北京: 机械工业出版社, 2000

2. Garcia-Molina H, Ullman D, Widom J 著. 数据库系统实现. 杨冬青, 唐世渭, 徐其钧等译. 北京: 机械工业出版社, 2001

(杨冬青)

bingfa moxing

并发模型 (models of concurrency) 描述并发系统行为的数学模型。

若一个系统内部发生的两个事件之间没有因果关系,则称此两个事件是并发的。因果关系不等于事件先后关系,有因果关系者必有先后关系,反之则不一定。存在并发事件的系统称为并发系统。例如操作系统是一个并发系统,人类社会也是一个并发系统。

并发概念由 C. A. Petri 于 1962 年首创。他的并发模型严格遵守并发即无因果联系的思想,用此模型描述的系统不含统一的时钟。此外还有另一种并发概念,例如 R. Milner 认为:若一个系统内部的两个事件可以按任意次序发生,则称此两个事件是并发的。习惯上称前一种并发为真并发,称后一种并发为交叠式并发。前一种并发概念的描述能力强于后一种并发概念的描述能力,但是在程序设计的大多数场合,后一种并发概念也够用了。

并发模型可分为两个层次:描述性的并发模型和语义性的并发模型。

描述性的并发模型 通常是一个形式系统,可以描述并发系统的行为。这种模型大体上有 4 类。

归约模型: λ 演算,项重写系统,图重写系统等都是。这类系统中的事件就是归约。它们缺少进程和通信的概念,不能用于分布式系统。

逻辑模型: 时序逻辑、动态逻辑、线性逻辑等都是。这类系统中的事件就是逻辑推导。它们能够在进程上作推理,但是缺少有效的通信手段,因此也不适用于分布式系统。

进程代数模型: 通信顺序进程 (CSP)、通信系统演算 (CCS) 等都是。这类系统以进程及进程间的通信为主要描述对象。系统中的事件就是进程通信,它们特别适合于描述分布式系统。

佩特里网模型: EN 系统,条件或事件系统,位置或变迁系统,有色佩特里网系统等都是。这类系统不直接描述进程和通信,但是可通过并发网络流表示进程和通信。系统中的事件就是事件节点的点火。这类模型不仅用于描述程序系统,也用于描述社会系统。

语义性的并发模型 第一类并发模型通常以指称语义中的幂域理论为基础。第二类并发模型常采用 Kripke 的可能世界理论。第三类并发模型也采用幂域理论,但采用不同的对象作为域的元素。例如针对同一个 CSP, M. Broy 给出的流语义以可能是

无限长的动作序列(称为流)为域元素,而 C. A. R. Hoare 等人给出的失败语义则以进程史 (s, X) 为域元素,其中 s 是一个有限动作序列,称为进程史的踪迹, X 是一个有限动作集,称为进程史的拒绝集。第四类并发模型的语义有多种表示方法。这里列举三种。第一,变换语义学方法,采用形式化的手段把高层次的佩特里网变成低层次的佩特里网,以使用后者的语义描述前者的语义。但是这种方法不能解决低层次佩特里网的语义描述问题,因之是不彻底的。也可以把佩特里网变成其他的并发模型,然后再利用其他模型的语义来描述佩特里网的语义。第二,行为语义学方法,用佩特里网点火的记录来表示它的语义,其中的每个点火都可以是多个事件节点的并发点火,一个佩特里网的所有可能的有限点火序列构成一个佩特里语言。这种方法有点像进程代数模型中的流语义。第三,进程结构方法。按进程展开一个佩特里网,即得到一个类似于黎曼曲面的多层网。这种结构蕴含了该佩特里网的所有可能进程,可以作为佩特里网的语义。

每一种并发模型被提出时,通常都附有一种给定的语义,或者是一开始没有明确的语义,后来有人赋予它一种语义。例如, R. Milner 为 CCS 配备了交叠式语义,而 C. A. Petri 则为佩特里网规定了真并发语义。但这不等于说任何并发模型天生注定只能有一种语义。例如,利用多层佩特里网方法,同样可以为 CCS 建立一个完整的真并发语义。

并发模型的性质 作为一个形式系统,并发模型有许多重要的性质值得研究。这些性质大体上都是以进程为中心的。

通信: 进程间交换数据称为通信,数据的发送和接收同时发生的称为同步通信,否则称为异步通信。

同步: 各进程所执行的动作次序的互相制约称为同步。进程间的同步往往通过通信来实现。在佩特里网中用同步距离表示制约的严格程度。

死锁: 多个进程为争夺同一资源(如设备使用权)而相持不下(谁也不能执行)称为死锁。

矛盾: 多个进程为争夺同一资源而产生的互斥现象(只能有一个进程执行)称为矛盾。

活锁: 由于资源使用权分配不当而使某个进程一直处于等待状态称为活锁。不存在活锁的系统称为是公平的。

活性: 当前有动作可执行的进程称为是活动进程。任何时候都含有活动进程的系统称为是有活性

的系统。

可观察：如果能从一个进程的外部测试出该进程执行了某个动作，则称此动作是可观察的。在 CCS 中，进程间的通信是不能从外部观察的。

发散：如果一个具有活性的系统所执行的任何动作都是不可观察的，则称此系统是发散的。

参考文献

1. Petri C. A. Kommunikation mit Automaten. PhD Thesis, University Darmstadt, 1962
2. Milner R. A Calculus of Communicating Systems. LNCS 92, Springer Verlag, 1980
3. Hoare C. A. R. Communicating Sequential Processes. Prentice-Hall, 1985 (陆汝钊)

bingxing bianyi chengxu

并行编译程序 (parallelizing compiler) 处理并行语言的编译程序或串行语言的程序并行化的编译程序 (自动并行编译程序)。

自动并行编译的实现在常规编译程序的基础上采取两种途径：预处理方式，在词法语法分析前进行；中间代码优化，在中间代码的基础上进行。预处理的方法，对数据依赖及循环中的数组下标分析清晰、精确，但并不适用于多种语言而导致重复劳动；中间代码的优化方法可通用化，但由于各语言实现时的中间代码形式各异且不对外开放，因而仅适用于系统制造者内部实现。自动并行编译的主要内容是并行性的识别。其过程是：数据依赖分析，即识别各种依赖如数据依赖、控制依赖等；程序转换，主要是循环的转换；借助运行系统和操作系统的支撑将源程序转化为并行代码或并行程序。自动并行编译一般进行源到源转换，即将串行源代码转换为并行源代码。

并行语言的编译程序通常由词法语法分析、优化和代码生成三个阶段组成。其中优化是主体，它包括依赖关系分析、循环转换及进程分配、调度、同步和通信等。

并行编译程序的主要研究内容包括：

(1) 依赖关系分析

依赖关系主要有四种：流依赖、反依赖、输出依赖及控制依赖。其中只有流依赖是固有依赖，其他依赖如输出依赖和反依赖在一定条件下可通过换名扩张及设置必要的附加语句消去，而控制依赖亦可转化为数据依赖。循环的并行化主要是通过依赖关系分析，改变原串行程序的词法次序，以缩短程序的

执行时间而不改变其语义。因此，依赖关系分析提供了保证串行程序语义正确性的条件。

编译程序依赖关系的测试方法主要有两种：精确测试法和近似测试法。精确测试法试图求出丢番图方程的通解以适合于方程数目较少且循环具有确定的上下界的情况，如依赖凸边域 DCH 方法。近似测试法仅给出一些存在解的必要条件，如 Banerjee 测试法、GCD 测试法及 λ 测试法等。

(2) 循环转换

循环并行化之前应进行预优化工作，这包括循环的规范化（尽可能线性化以便于依赖关系的分析，使循环的上界和步长均为 1 等）。

循环的并行化可分为三个阶段：依赖分析（跨迭代和迭代内依赖）、循环转换和循环调度。循环的转换技术是对循环变量的集合进行划分或改变循环，主要技术有循环交换、合并、分布、分片和环路收缩等。循环调度则是为了取得更好的负载均衡，可分为静态和动态两类调度方法，这样的技术有：自调度 SS、块调度 CSS、引导调度 GSS 和梯形调度 TSS 等。

(3) 过程分析

过程分析主要涉及别名分析、常数分析和引用分析，其目的是为了发掘被过程所隐蔽的信息以利依赖分析。对过程调用的处理，可在程序优化之前，采用内联扩展的办法即将被调用过程通过参数代换嵌入到调用处，最终使整个程序只包含惟一的过程，这种方法简单有效，但极耗内存，是一种消极的方法。已提出的过程分析技术有数组线性化、线性不等式、原子图及基于简单域的区域分析 RSD、RRSD 和 DAD 等。

自动并行编译工作最早始于 20 世纪 70 年代 CRAY-1 上的向量处理，当时只能识别某些适合于向量化的模式。70 年代末，美国 Illinois 大学研制了 Paraphrase，在向量化和并行化领域进行了开拓性的研究。此后这方面的工作相继展开，及至 80 年代，串行程序的自动并行化技术（围绕 FORTRAN）取得了长足的进步，其主要标志是数据依赖关系的分析技术的成熟、商品化工具（如 KAP 等）的出现及运用这些产品的系统开始获益。Illinois 大学的 U. Banerjee, M. J. Wolfe 和 Rice 大学的 K. Kennedy, R. Allen 等对数据依赖的分析和循环转换的研究，为向量化及后来的自动并行化工作奠定了理论基础。

由于现有程序中某些算法的固有顺序性，完全

自动化的并行编译十分困难。事实上,一个性能良好的并行程序可能需要设计合适的并行算法,而不仅仅是并行性识别或代码转换。所以,增加并行语言成分和自动并行化的结合可能是今后发展的趋势。

参考文献

1. Banerjee U. Dependence Analysis for Super Computing. Boston: Kluwer Academic Publishers, 1988
2. Ferrante J, Ottenstein and Warren J D. The Program Dependence Graph and Its Use in Optimization. ACM Trans. on Programming Languages and Systems, 1987, 9(3): 319 ~ 349
3. Allen R, Kennedy K, Porterfield C and Warren J. Conversion of Control Dependence to Data Dependence. Proc. of the Symposium on Principles of Programming Languages, 1983, 177 ~ 189
4. Wolfe M J. Optimizing Supercompilers for Supercomputers, Ph D. Thesis, University of Illinois at Urbana-champaign, 1986 (谢立 朱根江)

bingxing chengxu sheji

并行程序设计 (parallel programming) 一种适应于并行处理系统的程序设计方法。由于并行处理计算机系统有两种类型:单指令多数据流 (SIMD) 和多指令多数据流 (MIMD), 并行程序设计也有两种类型。

早期的并行系统是上述第一类并行系统。并行程序设计的主要方式是把可并行处理的操作数用向量表示,通过计算机提供的向量运算达到并行计算的目的。例如进行两个向量相加的运算:

$$A = (a_1, a_2, \dots, a_n)$$

$$B = (b_1, b_2, \dots, b_n)$$

求 $A + B$ 。

在并行程序设计的方式下求 $A + B$ 是一个并行操作,该操作将这 $2n$ 个数同时进行 n 次加法运算而得到 $A + B$,然后将结果的 n 个数同时存入 n 个单元而得结果。在实际计算问题中操作数往往不是如上所述已明显地表示成向量的形式,必须经过“向量化”以后才可进行并行操作。例如对 8 个数求和:

$$C = \sum_{i=1}^8 a_i$$

采用串行程序设计时要进行以下 9 个操作:

$a1 \rightarrow L$ (L 为累加器)

$L + a2 \rightarrow L$

.....

$L + a8 \rightarrow L$

$L \rightarrow C$

采用并行程序设计时,首先将操作数向量化,先计算

$a1 \quad a2 \quad L1$

$a3 \quad a4 \quad L2$

$a5 + a6 \rightarrow L3$

$a7 \quad a8 \quad L4$

再进行

$L1 \quad L2 \quad L1$

$+ \rightarrow$

$L3 \quad L4 \quad L2$

$L1 + L2 \rightarrow L$

$L \rightarrow C$

一共只要 4 个操作。

对于第二类并行处理系统的并行程序设计可分为批处理并行和单作业并行。前者指的是一批作业在计算机系统中被并行地处理,其中每个作业可能是串行地被处理,后者指的是一个作业分成几个独立的部分,它们分布在各个处理机上并行地被处理,通常研究的并行程序设计是指后者。在并行程序设计中的关键问题是如何将一个作业分解成可并行处理的部分来并行地处理。作业的分解一般有两种方法,一是人工分解,用户用显式并行程序设计语言来书写程序,写出的程序明确地被分解为可并行执行的部分;另一是自动分解,由编译程序在编译时分解。人工分解实现容易,但对程序员有较高的要求,特别如果要将在串行程序设计语言编写的程序移植到并行系统时,它们必须用新的并行程序设计语言重写程序,这往往是难以办到的。自动分解能解决这一问题,但实现自动分解难度较大。目前这两种方式都在研究。在研究作业分解时切不可认为分得越细越好。分得细,称为粒度小,并行程度高,但各部分之间的通信开销大。粒度大,并行程度低,但是通信开销小。如何控制粒度使得并行效率达到最高也是当前研究的一个难题。

用以进行并行程序设计的程序设计语言称为**并行程序设计语言**,它可分为显式并行语言和具有并行编译功能的串行语言。
(孙钟秀)

bingxing chengxu sheji yuyan

并行程序设计语言 (parallel programming language)

一种用于并行程序设计的语言。并行程序设计语言可分为显式并行语言和具有并行编译功能的串行语言。显式并行程序设计语言可以用传统串行语言加上并行语句等扩充的办法形成,也可以设计一个全新的具有并行功能的语言。这种语言有 SISAL, FORCE, LINDA, PARLOG 和 PCF FORTRAN 等等。具有并行编译功能的串行语言从用户使用角度看是一个传统的串行语言,但它的编译程序可将程序分解为并行执行的部分。

使用显式并行语言进行程序设计时用户要解决以下三个问题:①逻辑分解,即寻找一种适应并行处理的代码和数据划分。②从逻辑分解到处理系统的映射,即从资源分配负载均衡等考虑各程序部分怎样分布在系统的各台处理机上。③数据的定位。虽然上述三个问题很难,但是如果程序员的水平较高,采用显式并行语言可达较高的并行度从而较高地提高系统的效率。

并行编译的过程可分为三个阶段:词法和语法分析,优化以及并行代码生成。优化是并行编译的主体,它包括以下三部分:依赖关系分析,识别;程序转换,主要是循环转换;进程的分配及调度。20 世纪 70 年代末美国 Illinois 大学首先开展了向量化和并行化的工作。随之出现了许多 FORTRAN 向量化、并行化的工具,为后面的工作奠定了基础。80 年代末并行化的工作已较多地开展起来。如 APC FORTRAN,交互式并行化工具 PTOOL 和 PAT 等。90 年代以来并行处理技术已成为计算机的一种关键技术,并行程序设计语言将会有较大的发展。

(孙钟秀)

bingxing chuli xitong

并行处理系统 (parallel processing system)

利用多个功能部件或多个处理机同时工作来提高系统性能或可靠性的计算机系统。

任何一个计算机系统都包含某种程度的并行性,但如果只具有硬件基本操作的并行性,如一个数据的所有位同时传送,许多门电路同时工作等,不能认为是并行处理系统。并行处理系统至少应包含指令级或指令级以上的并行。20 世纪 70 年代的流水线向量计算机在当时被认为是典型的并行处理系统,但后来用基于流水线技术的 RISC(参见精简指

令集计算机)处理器构成的单机工作站,即使带不少外部设备和终端,一般也不认为是并行处理系统。所谓并行处理系统主要是指并行计算机系统或多处理机系统。

并行处理系统可以在 4 个级别上实现并行处理:指令内部、指令之间、任务或过程(程序段)之间和作业或程序之间。采用多个功能单元并行实现一条指令中的不同操作属于指令内部并行,超长指令字(VLIW)计算机(参见多发射结构)是实现指令内部并行的典型例子。同一时间执行两条以上指令称为指令间并行,超标量计算机(参见多发射结构)中有多条指令流水线,这是指令间并行的实例。一个程序往往可以分解成多个任务、子程序或过程,同一程序内多个任务或过程可以在一个系统的不同处理机中同时运行,以缩短计算时间,称为任务级并行。多个作业或大型计算问题的多个独立的程序,在并行处理系统的不同的处理机或计算机中同时运行,以提高系统的吞吐量或有效地利用系统资源,称为作业级并行。

并行处理系统的研究与发展涉及计算理论、算法、计算机体系结构、硬件、软件(包括操作系统、编译、编程环境与程序语言等)以及性能评价等方面。并行处理系统与分布式处理系统有密切关系,随着数字通信技术的不断发展,两者的界限越来越模糊。从广义上讲,分布式处理(参见分布式处理系统)也可以认为是一种并行处理形式。

发展历程

20 世纪 40 年代冯·诺依曼提出的细胞自动机是并行计算机的一种理论模型。1958 年美国国家标准局研制的 PILOT 计算机由 3 台独立的处理机构成,可以简单地协同工作。在 60 年代和 70 年代,多功能单元计算机、阵列处理机和流水线向量计算机陆续从实验室走向市场,其中影响较大的并行计算机系统有 CDC 6600, IBM 360/91, Illiac IV, TI ASC, CRAY-1 等。在此期间也开展了多处理机研究,如美国卡耐基·梅隆大学研制的 C.mmp 和 C*m 等。80 年代是向量计算机成熟的年代,以流水线技术为基础的向量计算机成为巨型计算机的主流技术。大规模集成电路的飞速发展与并行处理技术的逐步成熟为并行处理系统的发展提供了技术基础。就并行处理而言,90 年代是多处理机时代,许多计算机公司推出共享存储多处理机系统或大规模并行处理机系统。并行处理的思想几乎与计算机同时诞生,但自电子计算机问世之后,经过半个世纪的努力还未

真正普及,这一方面是由于器件水平的限制;另一方面是人们的思维方式受串行处理的束缚,并行软件发展较慢。

从历史上看,并行处理系统沿着几条不同的途径平行地发展。在共享存储的并行处理系统方面,共享存储系统的研制起始于 70 年代卡耐基·梅隆大学的 C. mmp 项目,采用交叉开关连接 16 个 PDP 11/40 和 16 个存储器模块。80 年代纽约大学研制的 Ultracomputer 和伊利诺依大学研制的 Cedar 分别对多处理机的互联网、同步、并行编译以及性能评价等做出过重要贡献。90 年代斯坦福大学的 DASH 和 FLASH 项目推动了分布式共享存储多处理机的研究。在消息传递并行处理系统方面,80 年代初起始于加州理工学院的 Cosmic Cube 项目,后来发展成为 Intel 公司的 iPSC 并行机和 Paragon 大规模并行计算机。90 年代初采用超立方体结构的 Ncube 并行计算机和麻省理工学院采用三维网格结构的 J-machine 实验系统对消息传递技术的发展也做出了贡献。在向量计算机方面,70 年代初 CDC 公司推出了世界上第一台向量双机系统 CDC 7600,后来向量计算机的体系结构分成寄存器到寄存器与存储器到存储器两个分支,Cray 公司沿前一支先后推出 Cray-1, Cray-X/MP 和 Cray-Y/MP,成为巨型计算机的主要供应厂商,日本的 Fujitsu, NEC, Hitachi 等公司也先后推出过性能很高的向量处理机系统。CDC 公司与 ETA 公司采用存储器到存储器结构,80 年代曾推出过 Cyber 205 和 ETA 10 等向量机,到 90 年代这一分支已不再发展。在阵列处理机方面,60 年代伊利诺依大学研制的 Illiac IV,80 年代 Goodyear 公司采用位片处理机构成的 MPP 和 90 年代初 TMC 公司的 CM-2 是阵列处理机的代表产品。在多线程并行处理系统方面,多线程并行处理的思想起源于 CDC 6600 的多功能单元,最早实现多线程处理的并行计算机是 HEP,以后 Tera 并行计算机与麻省理工学院研制的 Alewife 计算机也发展了多线程技术。在数据流计算机(参见数据流计算机)方面,自从 70 年代初 J. Dennis 提出数据流计算机概念以后,数据流计算机的研究一直没有中断,除了麻省理工学院的 Monsoon 和 *T 项目以外,英国、日本也先后研制了 Manchester 机、Sigma 系列以及 EM 系列数据流计算机。

并行计算机分类

由于计算机技术十分复杂,很难用一种观点对并行计算机做精确的分类。较普遍采用的是 M. J.

Flynn 于 1966 年提出的分类方法,即按指令流与数据流为 1 个或多个将计算机分为 4 类:单指令流单数据流(SISD)、单指令流多数据流(SIMD)、多指令流单数据流(MISD)和多指令流多数据流(MIMD)计算机。所谓指令流是指计算机执行的指令序列,数据流是由指令流所调用的数据序列。根据上述分类,除 SISD 计算机以外,其余 3 类都属于并行计算机。但是,很难设想一台计算机的多个指令流可以加工同一数据流,因此,多数人认为 MISD 计算机实际上不能实现,也有人认为将同一数据流以流水线方式进入由多个处理机构成的脉动阵列是 MISD 计算机。由于对数据流与指令流的理解不一样,一种计算机可能划归不同的类型。例如,最初 Flynn 认为流水线计算机属于 SISD 计算机,后来他又将流水线计算机归类于 SIMD 计算机。

单指令流多数据流计算机 SIMD 计算机的一种类型是**阵列处理机**,它采用一个控制单元控制许多处理单元,每个处理单元同步地执行同一指令流。由于每个处理单元的数据相互独立,可采取数据并行方式工作。处理单元的数目可成千上万,甚至上百万个。阵列处理机适合做大型数组运算,专用性较强。Maspar 公司的 MP-1 和 MP-2 是商品化 SIMD 阵列处理机的代表。

向量计算机是另一类 SIMD 计算机。它采用 1 条或多条流水线,在标量处理机上附加很强的向量处理能力,特别适于做大型科学工程计算。一个向量中的各个元素都要进行同样的运算,一个运算(如加法或乘法)可以分成若干个步骤,将执行不同运算步骤的功能部件按先后次序连接形成一条流水线(参见**计算机流水线**),不等前一个向量元素运算完,下一个向量元素就可以进入流水线。流水线允许多个向量元素在同一时间间隔内执行同一种运算的不同步骤,即以时间上的重叠实现并行处理,从而提高计算速度。向量计算机往往采用很高的时钟频率,非常高速的器件和专门的散热技术,因而成本较高。Cray 公司的 Y-MP 和 C-90 是向量巨型计算机的代表。高性能的大型计算机系统也提供向量处理部件,如 VAX 9000 和 IBM 390/VF。

多指令流多数据流计算机 这种计算机由若干处理机或处理单元、存储模块和互联网组成,在每一时刻,不同的处理机执行不同的指令。MIMD 计算机包括具有单一地址空间的共享存储多处理机系统和具有多个地址空间的消息传递多计算机系统两大类。

共享存储多处理机系统 包含具有统一地址空间的共享存储器,各个处理机通过共享变量进行通信与同步,各处理机联系密切,实现高度的资源共享,因而又称为**紧密耦合并行处理系统**或**紧耦合并行计算机**。如果所有的处理机都有相同的权力访问外部设备,即每个处理机都可以运行操作系统、响应中断,没有主从之分,则称这种多处理机系统为**对称式多处理机(SMP)**系统。反之,如果只有1个主处理器执行操作系统,响应外设请求,其他的多个处理器只能运行用户程序,这种结构的多处理机系统称为主从式多处理机系统。

多处理机系统的共享存储器可以集中在一起,通过高速总线或交叉开关与各个处理机相连,这种结构的技术较成熟,实现较容易。集中式共享存储的主要缺点是扩展性差,难以构成大规模并行系统。共享存储器也可以分布在各个处理机中,构成分布式共享存储多处理机系统,这种系统既具有共享存储编程容易等优点,又有很强的扩展性,但实现较困难,特别是保持高速缓存的一致性需要专门的机制(参见**共享存储**)。

消息传递多计算机系统 是一种**松散耦合并行处理系统**,其中每一个计算机都有自己的地址空间和局部存储器,通过**消息传递**方式进行相互通信。也就是说,这类并行计算机采用互联网将许多联系较松散的计算机组成一个并行处理系统,其中每个计算机(常称为**结点机**)都运行自己的操作系统,以消息传递方式实现结点机之间的同步。这种并行机的各个结点机一般以插件方式插在相对集中的1个或几个机柜里,并具有集中的控制台。

消息传递多计算机系统适于运行非常大的并行程序,其通用性不如紧密耦合的多处理机系统。它的主要优点是可扩展性强,能连接几千台甚至更多台计算机以构成大规模并行处理系统,性能价格比高,其缺点是编程较困难,软件工具有待进一步开发。

工作站簇 亦称**工作站机群**或**工作站网络**。实际上是一种分布式的并行处理系统。通信技术的迅速发展使得有可能利用标准的高速通信网络将多台工作站或高档微型计算机互连起来,构成一个并行处理系统。这种系统的通信方式与消息传递多计算机系统类似,其区别在于前者利用电信网络通信技术,如**异步传送模式**等。研制这类并行处理系统的关键技术在于降低进程级通信的软件开销,即高效率的通信接口。

除上述分类之外,从用户使用的角度可将并行计算机粗略地分成细粒度、中粒度、粗粒度3大类。并行计算粒度表示并行处理系统中的处理机独立执行的基本程序段的大小,即平均执行多少条指令后必须与其他处理机通信及同步。细粒度并行计算机一般由大量较简单的处理机组成,大都是按同步方式工作的SIMD计算机。中、粗粒度并行计算机大都是按异步方式工作的MIMD计算机。并行计算粒度反映处理机的处理能力与通信能力的比值。一般,粒度越细,并行化程度越高,但相应的通信与同步开销也越大。共享存储多处理机系统常用于支持细粒度或中粒度并行,而消息传递多计算机系统则多用于支持粗粒度或中粒度并行。

关键技术

并行处理与串行处理最主要的区别是并行工作的各个处理机或处理单元需要互相通信并实现必要的同步,以及一个任务需要分解并分配到各个处理机中协调地执行。因此,构造并行处理系统的关键技术包括处理机的互连、处理机之间的通信与同步以及处理机的调度策略。

互联网是区分不同并行处理系统的重要特征。较普遍采用的互联网包括多处理机系统总线、交叉开关、多级互联网以及点到点的静态互联网,如网格、超立方体以及胖树网络等。

存取非本地信息的通信延迟和保证并行计算结果正确性所付出的同步开销是影响并行计算机性能的主要因素。对于具有统一地址空间的共享存储系统,尤其是分布式共享存储系统,保持高速缓存一致性或设计适合并行的存取模式是最关键的技术。对于基于消息传递机制的松散耦合多计算机系统,消除结点机与互联网接口上的通信瓶颈,实现通信与计算的重叠是提高性能的关键。实现并行处理系统的通信与同步需要在硬件支持与软件支持两方面折衷考虑,直接用硬件支持可获得较高性能,但实现较困难,成本高。采用软件实现时又要在系统软件与用户软件之间折衷选择,在用户层进行通信,不进入操作系统,开销会大大降低,但通用性差,也加重了用户负担。

并行计算机的效率在很大程度上取决于并行系统软件。并行计算机能否普遍推广的关键在于能否提供方便而有效的并行应用软件以及友好的人机界面。并行计算机的系统软件包括并行操作系统、并行语言、并行编译和并行编程环境。并行操作系统的一项重要任务是对多个处理机进行自动的任务调

度,使互不相关的资源同时工作。操作系统核心的并行化可采用以下3种办法:主从式、浮动执行与多线程机制。多线程机制可以最大限度地开发核心代码的并行性。并行操作系统大都是在UNIX基础上进行扩充而形成的。

并行编程有多种模型,包括共享变量、消息传递、数据并行、面向对象、函数式与逻辑程序等。设计并行语言要么抛开现有的串行语言,要么在现有的串行语言上扩展并行功能。重新设计的优点是不受过去语言中串行思想的束缚,但与用户熟悉的语言不兼容。因此并行语言多采用扩充并行功能方式,保持与FORTRAN和C等语言的兼容。并行语言的编译也有几种不同做法,如编译制导或宏处理,对扩展部分做预编译处理以及自动化的并行编译等。用户最满意的编译器是能自动检测并行性并把串行程序转换成并行程序的优化重构编译,FORTRAN语言的自动并行已有一些成果,C语言也有一些并行库,但大部分并行程序还是靠程序员重写或人机结合半自动完成。除了语言和编译系统,并行编程环境还包括各种软件编程工具与支撑系统,其中最重要的是方便而有效的并行程序调试工具。编程环境是并行计算机与用户的界面,应使用户使用并行计算机像使用微型计算机、工作站那样方便。

应用与发展前景

由于信号在计算机中传播速度不能超过光速,而微电子工艺又受量子效应等物理规律的限制,半导体器件的开关速度与集成度已接近其物理极限,提高计算机系统的处理能力主要将依靠并行处理技术。并行处理已成为计算机领域中最受重视的研究领域之一。

并行计算机广泛用于大型科学、工程计算和大型事务处理。几乎所有的工作站与小型机厂家都生产了共享存储多处理机服务器,且已在金融、财贸、通信、交通、政府以及企事业单位管理等各个领域发挥了重要作用。并行计算机也已进入大学、科研单位和各个计算中心。其应用的典型实例包括数值天气预报、空气动力学计算(例如模拟风洞实验)、海洋动力学与天体物理计算、石油勘探、核反应模拟、基因分析与遗传工程研究、模式识别与人工神经网络计算、VLSI芯片设计、化学反应速度预测以及有限元分析等。

巨型向量计算机的计算速度已达每秒几百亿次浮点运算。以微处理机为基础的大规模并行计算机比传统的向量计算机性能价格比高,将以更快的速

度发展。大规模并行计算机中的处理机数量已达数千台甚至更多,计算速度已突破每秒万亿次。今后,大规模并行处理系统(包括具有数百万处理单元的SIMD计算机系统)将向每秒 10^{15} 次浮点运算的目标发展。商品化的共享存储多处理机已达到每秒几百亿次浮点运算的计算能力,性能上已达到巨型计算机的水平,但成本比传统的巨型计算机低一个数量级。由于共享存储多处理机系统通用性强,将会更广泛地普及。随着并行编译技术和并行编程环境的逐步成熟,具有分布式共享存储的MIMD计算机和以标准通信网为基础的工作站簇将成为并行计算机的主要产品。

参考文献

1. Bell G. Ultracomputers: A Teraflops before Its Time. Communications of the ACM, 1992, 35(8): 27~47
2. Denning P J, Tichy W F. Highly Parallel Computation. Science, 1990, 250(11): 1217~1222
3. Hwang K. Advanced Computer Architecture: Parallelism, Scalability, Programmability. New York: McGraw-Hill, 1993

(李国杰)

bingxing fangzhen

并行仿真(parallel simulation) 两个以上独立处理机同时或并发地执行仿真程序,以达到高速完成同一仿真任务的过程和方法。

随着仿真研究的对象日益复杂,规模日益增大,尽管计算机技术突飞猛进,但仍然满足不了仿真的要求。人们分析过,单中央处理器的数字计算机的计算速度不可能超越电路信号传输速度的极限,即电场传播速度 3×10^8 m/s。如果传输线按1m计算,则计算机的反应速度不可能超过 3×10^8 次/s操作。然而,三自由度空间飞行器的实时仿真要求大于 10^7 次/s操作,一个核反应堆的仿真要求大于 10^{10} 次/s操作,而大系统的优化研究则要求大于 10^{12} 次/s操作等。

仿真技术的发展依赖于计算机技术的进步,同时由于复杂系统对仿真技术的需求也促进了计算机技术的进步。于是,并行处理技术的研究及多机并发处理系统得到了迅速的发展。

多机并行仿真系统

根据多机并行仿真系统的结构、并行级别、软件体系等特征,我们可以将其进行分类。

按系统结构分类 根据多处理机构成系统时的

耦合紧密程度,可以将多机并行处理系统分为:分布式结构系统、松连接系统、紧耦合系统。

(1) 分布式结构系统 指构成系统的每一个处理机均拥有自己的存储器、接口、外设及操作系统,机间通信通过计算机网络或一般接口来实现。典型的是近些年来得到迅速发展的分布交互式仿真(DIS)系统。

(2) 松连接系统 在该系统中每一个处理机均拥有自己的内存、接口、外设及操作系统,这一点与分布式结构系统相同,但机间通信采用高速总线交换信息和共享外存。各类簇(cluster)系统是一种典型结构。

(3) 紧耦合系统 系统中所有处理机共用一个存储系统,系统内所有处理机均能平等地占用所有硬件资源,由统一的操作系统协调控制各处理机的操作同步和信息交换。这是真正典型的并行处理系统,如美国并行计算机公司生产的 PE3200 MPS,SGI 生产的 ONYX 系统等。

按并行级别分类 根据级别,可分为计算机级、处理机级、中央处理器(CPU)级三类。

(1) 计算机级并行系统 又称为多计算机系统。各并行工作单元是一台包括处理器、存储器、外部设备的完整的计算机,有自己独立的操作系统,各自执行一个完整的任务,各单元可独立工作。

(2) 处理机级并行系统 最典型的是阵列处理机系统。各并行处理单元是包含存储器和输入、输出(I/O)接口的计算器。处理器排成阵列,在同一控制器控制下执行相同的指令。

(3) CPU 级并行系统 它是单处理器的功能扩充,增加协调处理器、浮点处理器、函数运算器等,所以也称为多处理器系统。

按软件体系分类 一般的计算机执行时,指令和数据是一条一条执行的,所以也称为单指令流单数据流(SISD)系统。并行仿真系统在指令执行级分为三类,即单指令流多数据流(SIMD)系统、多指令流单数据流(MISD)系统和多指令流多数据流(MIMD)系统。

单指令流多数据流系统即各处理机同时执行同一指令,但处理结果(数据流)却采用各自的通道。

多指令流单数据流系统的数据通过同一通道依次流过每个单元,各单元完成各自的工作,共同完成某一任务,这种系统也称为流水线处理机。

多指令流多数据流系统的各处理机一般担任一

个相对完整的工作,相互之间在某些时刻有信息联系(有些还要求同步)。这是功能并行处理系统,也是目前应用最广泛的并行处理系统。

多机并行仿真技术

尽管并行处理系统有各种不同的结构,因而产生了各种不同的并行处理技术,归纳起来,下述问题是所有并行处理系统中都必须加以解决的。

(1) 多机协调控制 多机并行协调的主要特征是并行运算和资源共享。随着处理机数目的增加,对资源的竞争以及由此产生的“死锁”和“排斥”问题日益严重。为解决这些问题,多机协调控制技术的有效性及其完备性需要解决。目前,时序调度法、优先级调度法、超时自限法、开锁关键法等方法在不同类型的并行处理系统中得到应用。

(2) 机间同步及通信 多机并行完成同一任务,机间要频繁地交换数据,而且往往需要同步。高速地完成多处理机之间的有效通信,减少通信等待时间,确保必需的同步,一方面在硬件设计时就需要加以充分的考虑,另一方面在算法及软件实现方面也要提供有力的支持。

(3) 并行算法与并行语言 为了适应多处理机并行计算的要求,必须有相应的并行算法,并要有并行语言加以实现。

(4) 并行操作系统 并行操作系统除了要完成系统资源管理外,还需进行任务分解分配、多机调度、同步及通信、解决总线等资源竞争,还能支持并行算法与并行语言,从而为用户提供实现多机并行处理的环境。

参考文献

肖田元,张燕云,陈加栋. 系统仿真导论. 北京:清华大学出版社,2000 (肖田元)

bingxing gongcheng

并行工程(concurrent engineering) 集成地、并行地设计产品及其相关过程,包括制造过程和支持过程的系统化方法。这种方法要求产品开发人员从一开始就要考虑从产品概念形成到产品报废整个产品生存周期内的所有因素,包括质量、成本、进度和用户需求。它是一种先进的产品开发方式和制造模式。

并行工程是在制造业迫切需要进一步缩短产品开发周期、提高产品质量、降低开发成本、加强售后服务的背景下产生的。1982年,美国国防高级研究项目局(DARPA)开始研究如何在产品设计过程中

提高各活动之间“并行度”的方法。1986年, DARPA 发表了其研究成果, 奠定了并行工程的基础。1988年夏天, 美国国防部防御分析研究所(IDA)的 Winner 发表了著名的 R-338 报告, 明确提出了并行工程的概念和定义。从此, 并行工程逐步得到广泛认同。

并行工程的关键技术包括:

(1) 产品设计开发过程的重构和建模 即将传统的串行作业过程尽可能地转变为并行作业, 并进行过程建模。并行工程与顺序工程的对比如图 1 所示。

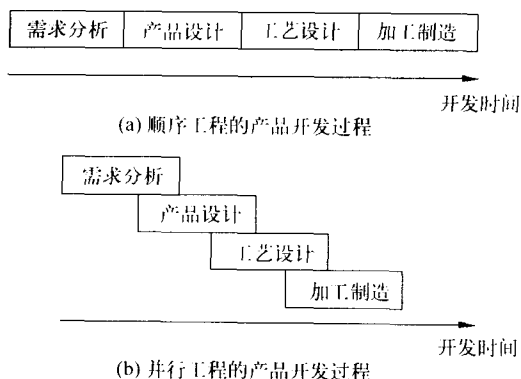


图 1 并行工程与顺序工程的对比

(2) 产品开发队伍重构 即将传统的部门制造专业组转变成以特定产品为主线, 支持并行作业的多学科协同工作小组。

(3) 集成化 CAX (计算机辅助设计 CAD, 计算机辅助工艺规划 CAPP, 计算机辅助制造 CAM, 计算机辅助实验 CAE) 和 DFX (面向装配的设计 DFA, 面向制造的设计 DFM, 面向质量的设计 DFQ, 面向成本的设计 DFC) 技术。

(4) 分布协同的产品开发环境。

并行工程的总体框架已趋于成熟, 并在一些企业得到了应用, 但它的若干关键技术如集成化 CAX/DFX 技术和分布协同的产品开发环境还远未成熟, 有待进一步研究。

参考文献

1. Winner R I, et al. The Role of Concurrent Engineering in Weapons System Acquisition. Institute for Defense Analysis Report R-338, Institute for Defense Analysis, December 1988
2. 唐荣锡. CAD/CAM 技术. 北京: 北京航空航天大学出版社, 1994
3. 熊光楞, 张和明, 徐文胜等. 并行工程的理

论与实践. 北京: 清华大学出版社, 2000

(高曙明 何发智)

bingxing shujuku

并行数据库 (parallel database) 以并行计算机为硬件环境并能充分发挥多处理和 I/O 并行性的数据库。

目前, 并行数据库以研究为主, 其研究主要围绕关系数据库进行, 集中研究如下 4 个方面:

(1) 实现数据库查询并行化的数据流方法

此种方法利用关系操作的固有并行性, 可以较为方便的对查询作并行处理。此种方法简单、有效, 目前已被很多并行数据库采用。

(2) 并行数据库的物理组织

此方法是研究如何把一个关系划分为多个子集合并将其分布到多个处理结点上 (称为数据库划分), 其目的是使并行数据库能并行地进行查询处理, 并能够进行读写多个磁盘, 充分发挥系统的 I/O 并行性。数据划分对于并行数据库的性能有很大影响, 目前数据划分方法主要有三种, 它们是一维数据划分、多维数据划分和传统物理存储结构的并行化。

(3) 新的并行数据操作算法

近年研究表明, 使用并行数据操作算法以实现查询并行处理可以充分地发挥多处理机并行性, 极大地提高系统查询处理的效率和能力。近年来许多并行算法已被提出, 主要是围绕 Join 操作的算法较多, 它们有基于嵌套循环的并行 Join 算法, 基于 Sort-Merge 的并行 Join 算法以及并行 Hash-Join 算法。

(4) 查询优化

查询优化不仅是传统数据库的重要组成部分, 也是并行数据库的重要组成部分。具有多个 Join 操作的复杂查询 (简称 MJ 查询) 的优化问题是查询优化的核心问题, 有关此项研究虽然刚开始, 但已取得不少可喜成果。

目前, 著名的并行数据库系统有: Arbore, Bubba, Gamma, Teradat 及 XPRS 等。随着并行计算机系统的发展, 并行数据库也将得到极大的发展。

(李建中 徐洁磐)

bingxing suanfa

并行算法 (parallel algorithm) 适用于并行

计算的算法。所谓并行计算通常由一些可同时执行的进程描述来表示,这些进程在执行过程中相互作用和协调工作,以完成对给定问题的求解。例如,1000个数相加的问题,将其分为4个进程,每个进程是一个部分和 $S_i (i=1,2,3,4)$,执行250个数相加。假设它们在由四台处理机 $P_i (i=1,2,3,4)$ 组成的并行机上进行计算,首先,每台处理机 P_i 执行一个进程,得到部分和 S_i ,当四台处理机完成了它们的部分和计算后,其他三台处理机(如 P_2, P_3, P_4)向 P_1 发送部分和 $S_i (i=2,3,4)$,然后, P_1 将部分和 S_1, S_2, S_3, S_4 串行相加,从而完成1000个数的并行计算。如不考虑通信开销,该并行算法执行速度比串行算法执行速度约提高4倍。并行算法性能的好坏直接影响并行计算机的使用效率,它是提高并行计算机处理效率的主要因素之一。

并行算法随着20世纪60年代并行计算机的研制而兴起,并随着并行计算机的发展而迅速发展。它的发展经历了几个重要阶段:①公理化阶段。60年代并行算法的研究建立在理想化并行计算机模型上,算法的复杂性度量建立在几条公理化假设上。②向量运算的并行算法设计阶段。70年代阵列处理机和向量处理机(如Cray-1, YH-1)相继研制成功并投入使用,并行算法研究进入实践阶段,研究内容主要基于流水线向量处理机的向量计算方法及其应用软件研制。70年代末和80年代初是其研究的顶峰时期,著名成果有递归问题的向量化。③多向量处理机的并行算法设计阶段。它是随着多向量处理并行机(如Cray Y-MP, YH-2)的出现而出现的,其特点是既要考虑多处理机间的任务级并行,又要考虑单处理机上的向量级细粒度并行。该类算法在80年代初期和中期比较流行。④多指令流多数据流(MIMD)类并行机上的并行算法设计阶段。该类并行机的显著特点是处理机功能比较强大,能独立处理各类复杂运算,处理机间通过显式或隐式的通信来相互协同,共同求解同一问题。该类并行机兴起于80年代初期,求解偏微分方程的区域分解算法是这方面的典型代表。⑤现代并行算法设计。目前,并行算法的研究主要集中于粗粒度MIMD并行算法,并要求具有可扩展性和易移植性,现代并行计算机的高性能获取要求并行算法设计兼顾两个方面:一是可扩展、易移植的粗粒度任务级并行;二是在每个进程,组织能充分发挥单机性能的合理数据结构、程序设计和通信方式。只有这样,才能真正发挥并行计算机潜在的性能。

1966年M. J. Flynn提出一种并行计算机分类方法,为并行算法的研究提供了两种并行计算机模型,即单指令流多数据流计算机(SIMD)和多指令流多数据流计算机(MIMD)。以此为依据,并行算法可粗略地分为同步并行算法和异步并行算法两类,每一类又包含数值并行算法和非数值并行算法两种。同步并行算法是指算法各进程的执行中,一些进程的执行必须等待另一些进程结果。异步并行算法是指算法各进程的执行均不需要相互等待,进程之间的通信是通过动态读、取存放于公用存储器中更新的信息或随机地相互使用对方送来信息的一类算法。数值并行算法基本上属于数值分析范畴,即对以数学形式表示的问题求其数值解。非数值并行算法基本上属于符号处理的范畴,即对以字符、数字、图形或其他记号表示的问题进行并行处理。

并行算法的研究分为三个层次:

(1) 并行算法理论 主要研究不同并行计算模型的能力、限制和等价关系、计算问题的可并行性和下界技术等。计算模型实际上是硬件和软件之间的桥梁,使用它能设计与分析算法及算法的实现。目前流行的并行计算模型有PRAM模型、BSP模型、LogP模型等。PRAM模型假定存在着一个容量无限大的共享存储器,有有限或无限多个功能相同的处理机,每个处理机具有简单的算术运算和逻辑判断功能,在任意时刻各处理机均可通过共享存储单元相互交换数据。BSP模型将并行机的特性抽象为三个定量参数 p, g, L ,分别对应于处理机数、带宽、全局同步之间的时间间隔。其计算由一系列全局同步分开的周期为 L 的超步组成,在各超步中,每个处理机执行局部计算,然后进行同步操作。LogP模型将并行机的特性抽象为四个定量参数 L (延迟)、 o (开销)、 g (连续发送或接收消息的最小时间间隔)和 P (处理机数或存储器数)。其中 L, o 和 g 三个参数刻画通信网络的特性,但却屏蔽了网络拓扑、路由算法和通信协议等具体细节。

(2) 并行算法的设计与分析 并行算法设计有三种方法:一是检测和开拓现有串行算法中的固有并行性,直接将其并行化。这样,算法的稳定性和收敛性等复杂问题,在直接并行化后就无需考虑。二是修改已有的并行算法,使之可求解另一类相似问题,称为“借用法”。此类方法不但要从问题求解方法的相似性方面仔细观察,寻求问题解法的共同点,而且借来的方法要用得上,效率高,从而达到借用的目的。三是从问题本身的描述出发,设计一种全新

的并行算法。通常包括平衡树法、倍增法、划分法、分治法、流水线法、加速级联和破对称方法等。

(3) 并程序设计和性能优化技术 常用的并程序程序设计有向量程序设计、共享存储并程序程序设计和消息传递并程序程序设计3种。向量程序设计以1991年FORTRAN 90语言的推出和许多自动向量化编译的研制成功为标志,已基本成熟。共享存储并程序程序设计以宏任务、微任务和自动任务技术为代表。1990年正式推出的PCF FORTRAN是共享存储并程序程序设计的语言文本。消息传递并程序程序设计是指用户必须通过显式发送和接收消息来实现处理机间的数据交换。每个并行进程均有自己独立的地址空间,相互之间的访问必须通过显式消息传递来实现,并行开销比较大,主要适合于粗粒度的并行计算。消息传递是当前并行计算领域的一个非常重要的并程序程序设计方式,支持当前所有的分布式存储、分布式共享存储、共享存储和NOWs/COWs系统。

并行编程环境以PVM、MPI、HPF、OpenMP为代表。并行虚拟机(PVM)是比较流行的基于消息传递的并行通信库,异构性和易移植性是其设计的主要目的。消息传递界面(MPI)于1994年推出,目的是实现消息传递库的标准化,以保证消息传递并程序程序设计的易移植性,已成为消息传递用户界面的工业标准。高性能FORTRAN在FORTRAN语言的基础上,提供一系列并行指导语句,用于描述并程序中数据的存储方式和指导循环级粗粒度并行。OpenMP于1998年推出,目的是实现共享存储并程序程序设计的标准化,以保证共享存储并程序的易移植性。目前,它在通常的程序设计语言的基础上,提供五类不同结构(并行区结构、工作共享结构、组合并行工作共享结构、同步结构和数据环境结构)的并行指导语句,以实现程序的循环级粗粒度并行。

程序性能优化技术主要有串行程序优化、共享存储环境下并程序优化和消息传递并行环境下并程序优化。串行程序优化技术有循环级优化、Cache命中率优化、过程级优化、Branch优化以及尽量采用标准库软件等。共享存储环境下并程序优化技术有循环级优化、高速缓冲存储器一致性优化、粗粒度并行优化等。消息传递并行环境下的并程序优化技术有减少通信开销、消除通信瓶颈和保持负载均衡等。负载均衡方法包括静态和动态负载均衡两种,其中静态是指并行模拟过程中,分配给各个处理机的任务是固定不变的,动态是指,必须根据各

处理机间负载平衡情况,动态调整各处理机的任务,以保证负载平衡。

并行算法的性能度量主要包括并行算法的运行时间、加速比与效率、并行算法的可扩展性等。并行算法的运行时间 t_p 是指求解一个问题的并行算法在 p 台处理机上从算法开始执行到算法执行完毕的这段时间。加速比(又称绝对加速比)定义为 $s_p = t_s/t_p$, t_s 是指求解相应问题的最快串行算法在单处理机上的运行时间。效率 $E_p = s_p/p$, p 是并行算法所需的处理机台数。由于求解一个问题的最快串行算法很难找到,甚至不存在。因此目前采用相对加速比进行度量,它定义为 $s_p = t_l/t_p$, t_l 是指求解相应问题的并行算法在一台处理机上的运行时间。可扩展性是指随着并行系统规模的增大,通过适当增加问题的规模,使并行系统的性能与其规模成线性比例增长。目前常用的有固定时间的加速比、等效率可扩展性度量、等速度可扩展性度量等。固定时间的加速比是指在保持问题规模与机器规模呈线性增长情况下度量加速比,如果加速比同机器规模趋于线性增长关系,则表明该系统是可扩的。等效率可扩展性度量是指系统规模增加时,测量增加多少运算量会保持效率不变。等速度可扩展性度量是指在保持系统的平均速度不变情况下考察问题规模与系统规模呈线性增长的能力。可扩展性不只依赖于机器规模和问题规模,也依赖于机器的存储器容量、I/O能力以及通信能力等因素。

为了使读者对并行算法有个基本的了解,下面举二个例子加以说明。

(1) 矩阵转置 在多个处理机构成的 $n \times n$ 网格上对 $n \times n$ 的矩阵 A 求转置矩阵 A^T 。设处理机 $P_{i,j}$ 存有矩阵元素 $a_{i,j}$,转置算法包含两个阶段,若将处于对角线上的处理机 $P_{i,i}$ 称为路由器,在第一阶段,每个对角线以上的处理机 $P_{i,j}(i < j)$ 将值 $a_{i,j}$ 往下以步进的方式传给处理机 $P_{j,j}$,到达 $P_{j,j}$ 需花费 $k = j - i$ 步。然后, $P_{j,j}$ 花费 $k = j - i$ 步将该值传送给 $P_{j,i}$ 处理机。第二阶段,所有对角线以下的处理机 $P_{i,j}(i > j)$ 将值 $a_{i,j}$ 向右以步进方式传给 $P_{i,i}$,然后 $P_{i,i}$ 将该值向上传送,以步进方式到达 $P_{j,i}$ 处理机。最终,每个处理机 $P_{i,j}$ 将含有值 $a_{j,i}$ 。转置算法可在 $O(n)$ 时间内完成,对于网格来说它是渐近最优的。

(2) 矩阵乘 设要计算2个 n 阶方阵 A 和 B 的乘积 C 。设 $p = m^2$ (m 为正整数),将 A 、 B 与 C 以适当方式分为 p 块 $A_{i,j}$ 、 $B_{i,j}$ 和 $C_{i,j}(0 < i, j \leq m)$,每块为 $(n/m) \times (n/m)$ 子矩阵,将这些块映射到 $m \times m$

个逻辑处理机阵列上,每个处理机记为 $P_{i,j}$ ($0 < i, j \leq m$), $P_{i,j}$ 存储 $A_{i,j}, B_{i,j}$, 并计算 $C_{i,j}$ 。在计算 $C_{i,j}$ 时,需要所有子矩阵块 $A_{i,k}, B_{k,j}$ ($0 < k \leq m$), 因此 A 的块在处理机阵列每行作多对多广播,而 B 的块在处理机阵列每列作多对多广播。当 $P_{i,j}$ 接收完 $A_{i,1}, A_{i,2}, \dots, A_{i,m}; B_{1,j}, B_{2,j}, \dots, B_{m,j}$ 后,执行子矩阵乘法和加法运算。在每台处理机上,计算时间为 $O(n^3/p)$, 而对应的串行时间为 $O(n^3)$ 。

参考文献

1. 李晓梅,莫则尧等.可扩展并行算法的设计与分析.北京:国防工业出版社,2000
2. 陈国良.并行计算——结构、算法、编程.北京:高等教育出版社,1999
3. Jack Dongarra, Ian Foster, Groffrey Fox, William Gropp, Ken Kennedy, Linda Torczon and Andy White. Sourcebook of Parallel Computing. Morgan Kaufmann Publishers, USA, 2003

(李晓梅 莫则尧 陈军)

bofen duolu fuyong

波分多路复用 (wavelength division multiplexing, WDM) 不同波长的光载波信号在同一根光纤中同时传输的**多路复用技术**。波分多路复用光纤通信系统分为单向系统和双向系统两种。单向系统使用合波器和分波器来实现波分复用功能。双向系统使用双向耦合器来实现波分复用功能。

在单向波分多路复用光纤通信系统中,发送端用**合波器**将工作在不同波长的光发射机所发射的光载波信号合起来,通过同一根光纤传送到接收端。在接收端用**分波器**将不同波长的光载波信号分开,然后分别将它们送到相应光波长的光接收机,对各自所接收到的光信号做进一步处理。在双向系统多路复用光纤通信系统中,采用**双向耦合器**将工作在不同波长的光发射机所发射的光载波信号合起来,通过同一根光纤传送到接收端。在接收端也用双向耦合器将不同波长的光载波信号分开,然后分别将它们送至相应光波长的光接收机,做进一步处理。由于通信两端通过一根光纤同时接收和发送,因此实现了双向波分复用功能。

复用器是合波器、分波器和双向耦合器的统称,它是波分复用光纤通信系统的关键。其主要性能指标是插入损耗和串扰衰减。**插入损耗**是指在波分多路复用系统中,由于使用了复用器而引起的附加损耗,一般要求插入损耗小于 $1.0 \sim 2.5$ dB。**串扰衰减**

是由于复用器件分光性能不完善,而在信道间产生一定程度的串音干扰。串扰衰减是指其他信道串入本信道的光功率与本信道所接收的有效信号光功率之比。

光波分复用技术在通信中具有很好的应用前景。光纤带宽高达 $2 \times 10^4 \sim 3 \times 10^4$ G 周,常规的光纤通信只利用了光纤带宽很小的一部分。如果能很好利用光波分复用技术,将可以使一根光纤的传输容量得以充分利用。光波分复用器件体积小、结构简单,信道传输具有全透明的特征。电子系统的速率在当今条件下已达到极限,但采用光波分复用则仍可大幅度提高光纤总传输速率。

光波分复用技术的发展包括光波分复用器件结构、性能的发展以及光波分复用技术在通信中的应用的发展。光波分复用器件结构、性能的发展趋势是:复用信道数增大,相邻信道间隔变小;使其具有灵活复用和分用的特征,即分波信道的波长可变;发展光集成波导,如平面光栅集成光波分复用器;采用新的工作机制,如用全息透射光栅制作分波器;开发全光纤特征的器件;研制高性能的激光器、光探测器、光纤等。光波分复用技术在通信领域中应用的发展速度也很快。通过掺铒光纤放大器和光波分复用技术相结合,能高效、经济地发挥长距离光纤网络的巨大带宽潜力,其潜在的传输容量至少比目前的系统增大 50 倍。并可大大提高网络的可靠性和灵活性。

(胡道元)

bofenghan

波峰焊 (wave-soldering) 利用熔融的焊锡波,将预先插上的所有元器件一次焊接到印制板上的一种工艺过程。波峰焊焊接方式一般采用机械泵或电磁泵,使熔融的铅锡合金焊料形成一种持续的焊锡波。焊接时将插好元器件的印制板装在传送框架上或搬送爪上,以恒定的速度连续地输送印制板,使印制板下面的焊盘全都接触到焊料,从而完成焊接的全过程。

波峰焊在焊通孔元件时,焊料润湿元器件引线突出的部分,由于毛细孔作用,焊料被吸入印制板的金属化孔里。平面安装型(SMD)元器件的焊接,则是将元器件先粘在印制板的焊接面,直接浸入焊料的波峰中。

整个波峰焊包括三道工序:助焊剂涂布、印制板预热、波峰焊接,参见图 1。

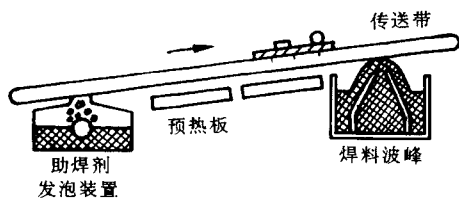


图1 波峰焊系统示意图

助焊剂涂布装置 助焊剂涂布装置用于在印制板上涂布助焊剂,其作用是除去锡波及焊接点的表面氧化物,使焊锡能在金属表面展开,降低溶锡表面张力,从而在元器件引线和印制板焊盘间形成良好的焊点。

助焊剂涂布主要采用以下两种方式。

发泡式 发泡式是以发泡石配合空压气体,形成发泡的助焊剂,当发泡助焊剂通过喷嘴喷射到印制板焊接面,气泡立即破裂,在板面沉积上一层薄而均匀的助焊剂层,多余的助焊剂则流回到焊剂槽里。板上助焊剂涂布厚度决定于焊剂的黏度,先进的波峰焊机均有焊剂比重自动监控装置,以控制焊剂的黏度。

喷雾式 有采用压缩空气和采用超声波两种方法。前者是在焊剂槽中放置一个旋转的滚筒,滚筒周围装有细孔金属网,压缩空气通入滚筒并从金属网的细孔中喷出,带动焊剂形成喷雾。后者是用超声振子使助焊剂雾化。

预热装置 常见的波峰焊机预热装置有电热管、石英管、电热板等方式,一般预热到 $80 \sim 120^{\circ}\text{C}$ 。对预热装置的要求是能在短时间内将印制板预热到所需要的温度,又能防止过度的热冲击;同时要使助焊剂汽化以便于清洗和加速助焊剂的化学反应,从而提高助焊性能。

波峰焊接 波峰焊接采用熔融的焊料。焊料由泵加压后经过一个通道,同时溢流到延伸板上,再回到焊料槽完成印制板的焊接。波峰焊设备主要有单波峰和双波峰两类。单波峰焊料与印制板仅有一个接触面,焊料波峰宽而且平稳;而双波峰焊料与印制板有两个接触面,两个接触面的大小和角度均不相同,以避免局部虚焊。

为适应平面安装型元器件的波峰焊的要求,已设计出多种新的锡波波形,以便在焊接过程中消除助焊剂和黏胶产生的气泡,消除一般稳定层流锡波易发生的阴影效应,避免因焊接时间太长使元器件因过热而损坏。新的波形有双波峰式、震荡式、气泡

式、喷射式等。

波峰焊比手工焊一致性好,可靠性高,但不适用于塑封有引线芯片(PLCC)以及间距 0.65 mm 以下的四列扁平封装(QFP)。(顾本斗)

Bolan jifa

波兰记法 (Polish notation) 完全省去了旨在改变运算次序的圆括号,而仅以前缀或后缀来描述表达式的方法。

1951年波兰逻辑学家 J. Lukasiewicz 提出一种逻辑运算无括号的记号,其具体做法是将表达式中的所有算符置于其操作数的前面或后面,其要求是每一个算符均有固定数目的操作数。无疑,这扩充了代数与其他运算-操作数系统。将算符置于操作数前面的这种前缀形式已在程序设计语言 APL 中使用了,而将算符放在操作数后面的这种后缀形式已用作许多编译程序的中间语言。后缀形式也称作逆波兰记法。

通常加号“+”是双目算符,它有两个操作数。因此,前缀形式“+ab”与“后缀形式“ab+”显然指同一个表达式“a+b”。类似地,逻辑运算非“¬”只指望有一个操作数。因此¬q的后缀形式是q¬。减号“-”会引起问题,这是因为它当作负号时,它指望有一个操作数,而将它当作减号时,则指望有两个操作数。为解决这一矛盾,只需将减运算改作加负数即可。如表达“a-b”改作“a+(-b)”,那么,它的前缀形式便是“+a-b”,而后缀形式便是“ab-+”。可见,前缀表示是一种表示(数学)表达式的方法,在表达式中,每个算符放在它的操作数的前面,并指明其后面的操作数或中间结果所要执行的运算。后缀表示也是一种表示(数学)表达式的方法,在表达式中,每个算符放在它的操作数的后面,并指明其前面的操作数或中间结果所要执行的运算。

如果前缀表达式从右到左求值,则每当扫视到一个算符时,其操作数便是最近刚求得的。如果后缀表达式从左到右求值,则每当扫视到一个算符时,其操作数也是最近刚求得的。考虑表达式 $(a+b)*(c-d)$,其前缀表达式是“*+ab-cd”,其求值序列是 $d, c, -cd, b, a, +ab, *$ 。上述表达式的后缀表示是“ab+cd-*”,其求值序列是 $a, b, ab+, cd, cd-, ab+cd-, *$ 。

波兰记法的优点在于表达式的求值不再受算符的优先规则所支配,也不使用诸如括号那样成对的

定界符。正是由于使用波兰记法得以使表达式求值容易并且表示惟一,所以它可用在计算机语言和语言处理程序(如编译程序)之中。

表 1 给出某些逻辑表达式的前缀表示与后缀表示。

除上面讲到的波兰记法例子外,作为编译程序的中间语言,有人还将它推广应用到语言的其他语法单位,比如赋值语句 $E := B$ 的后缀表示是“ $EB :=$ ”;条件语句 if B then S1 else S2 的后缀表示是“ $B S1 S2$ if-then-else”等等。显然, $:=$ 被看作是二个操作数的运算,而 if-then-else 被看作是三个操作数的运算。

表 1 一些逻辑表达式的表示

逻辑表达式	前缀表示	后缀表示
$a \leq b \wedge c > d$	$\Lambda \leq ab > cd$	$ab \leq cd > \Lambda$
$p \equiv q \vee r$	$\equiv p \vee qr$	$pqr \vee \equiv$
$p \equiv q \wedge r$	$\equiv p \wedge qr$	$pqr \wedge \equiv$

参考文献

1. Harms E, Zalinski M P. Introduction to APL and Computer Programming. New York: John Wiley and Son, Inc., 1977
2. Wilbur L R. Applied APL Programming. Prentice-Hall Inc., 1978 (段祥)

Bosite duiying wenti

波斯特对应问题 (Post's correspondence problem) 由 E. Post 1946 年提出的一个不可判定问题。

设 A 是非空字符有限集, A 中字符的有限序列称为 A 上的字符串。 A 上的 m 个字符串 u_1, \dots, u_m 的连接字符串, 记为 $u_1 \dots u_m$ 。 A 上的表是 A 上字符串的有限序列, 序列的长度称为表的长度。例如, ab, Λ, aa 是字符集 $\{a, b\}$ 上的长度为 3 的表, 其中 Λ 表示空串。

设 l_1, \dots, l_k 和 m_1, \dots, m_k 是 A 上两个表, 若存在小于或等于 k 的正整数 i_1, i_2, \dots, i_n 使 $l_{i_1} l_{i_2} \dots l_{i_n} = m_{i_1} m_{i_2} \dots m_{i_n}$, 则称表 l_1, l_2, \dots, l_k 和 m_1, \dots, m_k 有匹配。例如, $A = \{0, 1\}$, $l_1 = 1, l_2 = 10111, l_3 = 10, m_1 = 111, m_2 = 10, m_3 = 0$, 因 $l_2 l_1 l_3 = m_2 m_1 m_3 = 101111110$, 因此, l_1, l_2, l_3 和 m_1, m_2, m_3 有匹配。判定同一字符集上的任意两个相同长度的表有没有匹配的问题, 称为波斯特对应问题。

由于图灵机的停机问题是不可判定的, 可以推出波斯特对应问题也是不可判定的, 即不可能找到一个算法来判定同一字符集上的任意两个相同长度的表是否有匹配。

波斯特对应问题在形式语言理论和程序设计理论中有重要应用。由于波斯特对应问题是不可判定的, 可推出形式语言理论和程序设计理论中的许多问题是不可判定的。

参考文献

- Hopcroft J E, Ullman J D 原著. 自动机理论、语言和计算导引. 徐美瑞译. 北京: 科学出版社, 1986
(陈火旺 贵可荣)

Bosite ji

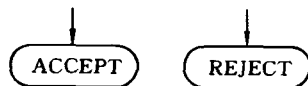
波斯特机 (Post machine) E. Post 提出的一种由表示宏操作指令的语句组成的计算模型。

在字母表 $\Sigma = \{a, b\}$ 上的一台波斯特机可以用一个带取值于 $\{a, b, \#\}$ 的字变量 x 的流程图来刻画。该流程图由下述语句组成:

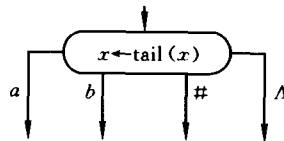
(1) 启动语句



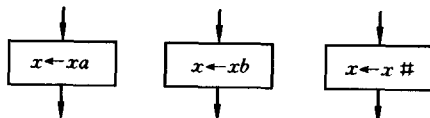
(2) 停机语句



(3) 分支判断语句



(4) 赋值语句



其中, Λ 表示空字 (长为 0 的字), $\text{tail}(x)$ 表示将字 x 的最左一个字母去掉后所获之字, xa 表示将字 x 右接上字母 a 后所获之字。分支判断语句将字 $\text{tail}(x)$ 赋给 x , 然后依据未赋值前的字 x 的最左字母而流向下一语句。在字母表 Σ 下的一个字 α 被

称为可被波斯特机 M 接受(拒绝)的,若该波斯特机在输入 $x = \alpha$ 时计算最终流到停机语句 ACCEPT (REJECT)。E. Post 证明了:在字母表 Σ 下的波斯特机器类与图灵机器类识别语言的能力是相同的。波斯特机作为计算数论函数来说,可计算一切图灵可计算函数,而且只需使用 $x := x + 1, x := x - 1$ 形的赋值语句和 GOTO $n, \text{ IF } x \neq 0$ 形的分支判断语句。(李祥)

Boerziman ji

玻耳兹曼机 (Boltzmann machine) 神经元间具有反馈作用的多层神经网络。它由输入层、输出层和隐层构成,网络没有明显的层次,神经元间相互连接,每一神经元可取 0 和 1 两种状态。在这种模型中,当神经元的输入加权和发生变化时,将引起神经元状态的变化和更新。这种更新在各个神经元之间是非同步的,可以通过概率统计方法来描述。由于网络的状态采用了统计物理学中的玻耳兹曼概率分布,因此称为玻耳兹曼机。考虑包含 n 个神经元的网络情形,第 i 个神经元的状态用 s_i 表示,它和第 j 个神经元之间的连接对称,即权值 $w_{ij} = w_{ji}$,没有自反馈。第 i 个神经元的状态 s_i 随机地被决定,其为 1 时的概率为 $p(s_i)$,满足统计决策规则:

$$p(s_i) = \frac{1}{1 + \exp(-\Delta E_i/T)} \quad (1)$$

其中, T 是温度参数。假如上述统计决策规则反复运用于每一神经元,网络将达到热力学平衡,即状态保持不变,状态的分布服从玻耳兹曼分布。玻耳兹曼机可看作 Hopfield 网络的一种推广,隐层的出现说明了玻耳兹曼机与前向多层网络的相似性。

玻耳兹曼机是 1985 年由 G. E. Hinton 等人利用了统计物理学的概念和方法发展起来的神经网络模型,可看作是一种外界概率分布的模拟机。其基本思想是将物理学中随机模拟退火的方法用于神经网络的状态分析。玻耳兹曼机的信息处理基于概率分布和统计动力学,它可实现任意随机函数的逼近和解决约束优化问题。

模拟退火算法将组合优化问题与统计力学中的热平衡问题类比,开辟了求解组合优化问题的新途径。它通过模拟退火过程,可找到全局(或近似)最优解。模拟退火算法是基于 Monte Carlo 迭代求解法的一种启发式随机搜索算法。在对固体物质进行退火处理时,通常先将它加温熔化,使其中的粒子可自由运动,然后随着温度的逐渐下降,粒子也逐渐形

成了低能态的晶格。若在凝结点附近的温度下降速率足够慢,则固体物质一定会形成最低能量的基态。对于组合优化问题来说,它也有这样类似的过程。组合优化问题解空间中的每一点都代表一个解,不同的解有着不同的代价函数值。所谓优化,就是在解空间中寻找代价函数(亦称目标函数)最小(或最大)的解。设 $S = \{s_1, \dots, s_n\}$ 为所有可能的组合(或状态)所构成的集合, $C: S \rightarrow R$ 为非负目标函数,即 $C(s_i) \geq 0$ 反映取状态 s_i 为解的代价,则组合优化问题就是找最小的代价函数 C 。模拟退火时把每种组合状态 s_i 看成某一物质系统的微观状态,而 $C(s_i)$ 看成该物质系统在状态 s_i 下的内能,并用控制参数 T 类比温度。让温度 T 从一个足够高的值慢慢下降,对每个 T ,用 Metropolis 抽样法在计算机上模拟该体系在此参数 T 下的热平衡态,即对当前状态 S 做随机扰动产生一个新状态 S' ,计算增量 $\Delta C' = C(S') - C(S)$,并以概率 $\exp(-\Delta C/kT)$ 接受 S' 作为新的当前状态。当重复地随机扰动足够多的次数后,状态 S 出现为当前状态的概率将服从玻耳兹曼分布。由此看出,模拟退火算法基本上由三部分组成:①以一定的概率密度跃迁到新的状态,这个概率密度函数称为生成函数;②以一定的概率密度容忍代价函数的偶然上升,这个概率密度函数称为容忍函数;③以一定的冷却程式降低温度,这个等效温度是生成函数和容忍函数中的控制参量,确定所引入的随机扰动(噪声)的强度。

玻耳兹曼机除了可以解决约束组合问题外,还可以通过学习,模拟外界所给出的概率分布,实现联想记忆。一种常见的玻耳兹曼机学习算法由下列步骤构成:①开始时设置高温 T ,并随机设置全部网络权值;②外加一个输入向量,用当前权值计算目标函数;③根据玻耳兹曼分布概率 $P(x) = \exp(-x^2/T^2)$,随机地改变每个权值;④重新计算目标函数,如果减少,则固定权值的改变,否则,根据步骤③玻耳兹曼分布决定的概率确定是否保持权值的改变;⑤用公式 $T(n+1) = T(0) (1/\lg(1+n))$ 计算新的温度值;⑥重复步骤③到⑤。在玻耳兹曼机的学习过程中,温度参数 T 是一个重要的参数。假如温度 T 低,网络只有很少几个可达到的状态,易陷入局部极小,学习非常困难;假如温度 T 高,网络可达到的状态多,状态易变换,但学习算法收敛的稳定性差。为了快速使网络达到低温平衡,玻耳兹曼机的学习一般采用模拟退火过程,先设置高温,然后让温度 T 逐渐从高温到低温减小。

玻耳兹曼机作为一种随机动力学系统,具有较强的信息处理能力,是处理带有随机扰动信息的一种有效方法,已应用于模式识别、语音识别和优化问题求解。但其主要的缺点是训练过程需要大量的计算,训练时间过长。为了加速玻耳兹曼机的学习过程,提出一些玻耳兹曼机模型的变形和学习算法的改进。改善玻耳兹曼机的行为,其中包括平均场理论和 H. Szu 提出的柯西 (Cauchy) 机等。玻耳兹曼机学习算法的改进,非对称结构玻耳兹曼机动力学行为的研究吸引着人们作进一步的探索。

参考文献

1. Ackley D H, Hinton G E, Sejnowski T A. Learning Algorithm for Boltzmann Machine. Cognitive Science, 1985, 9: 147 ~ 169

2. 史忠植. 神经计算. 北京: 电子工业出版社, 1993
(史忠植 张建)

boyishu sousuo

博弈树搜索 (game tree search) 一种具有双人完备信息特征的博弈问题求解方法。在这类博弈问题求解过程中,两位选手对弈,双方轮流走步,每一方不仅完全知道对方过去已走过的棋步,而且还能估计出对方未来可能的弈着。对弈的结果是一方赢(另一方输),或为和局。这类博弈的实例有:一字棋、西洋跳棋、中国象棋、围棋等。而带机遇性的博弈问题,如掷骰子和大多数扑克牌游戏,因不具有完备信息特征,所以都不属于这里讨论的范围。

博弈问题可以用产生式系统模型来描述。例如对国际象棋,综合数据库可用以描述棋盘上棋子位置的布局,产生式规则可用以描述各种棋子的合法走步。这些规则作用于数据库,产生出所谓的博弈图或博弈树。

博弈树是一类特殊的“与或”树。树的根节点代表博弈的初始状态,它的后继节点是对弈的第一位选手相对于初始状态可能走出的所有棋局。而它们的后继节点又是第二位选手相应可以走出的所有棋局,以此类推。博弈树的叶节点表示棋局的输、赢(一方的赢标志另一方的输)及和局。例如,设 MAX, MIN 二人对弈, MAX 先行。对于任意给定的当前棋局节点, MAX 可能有几种走法,即可以生成当前节点的若干后继节点,这些后继节点被称为“或”节点。对于 MAX 可能走出的每一个棋局, MIN 可能分别有若干种走法应付。对 MIN 的所有这些走法, MAX 必须有所准备。因此, MIN 的后继节点

被视为 MAX 的“与”节点。在此每个“与”节点之后又可继续下接 MAX 可能生成的数个“或”节点。“与”、“或”节点如此交替出现,直至达到标志胜、负、和局的叶节点,则一棵博弈树生成完毕。

在博弈树中搜寻使某方取胜的策略,类似于在“与或”图中搜索一个解图(参见 AO* 算法)。例如,若要使 MAX 取胜,那么在对博弈树从开局到终局的各层节点的搜索过程中,只需保证 MAX 对一个“或”节点取胜,但必须保证他对所有有关的“与”节点取胜。这是完全取胜的策略。实现这种策略就是在博弈树中搜索一个解图,解图代表了一种完整的博弈策略。这种寻找完整博弈策略的做法只适应于求解简单的博弈问题(如 Grundy 问题求解等),或收拾复杂博弈的残局。对于复杂的博弈问题,如中国象棋等,由于受时间和存储空间的限制,这种博弈搜索策略是不适用的,这时就需要采用有界深度极小极大博弈策略。

有界深度极小极大博弈策略的思想是:每当选手走步之前,可根据当前的棋局势态,预测出双方对弈的若干步数,并以此构造出一棵有界深度的局部博弈树,树的深度亦为所预测到的步数。然后,根据极小极大原则标记树中各棋局节点的势态优劣,并从可能的走步中选择一步相对最好的棋着。这种策略又称为“一步好棋”策略。策略中的极小极大原则是指:对手永远不会犯错误,他总是走出对自己最不利的棋着,自己则应从这种给定最坏的棋局中选择相对最好的棋着。复杂博弈问题的求解过程就是不断地重复执行上述构造局部博弈树,选择最好棋步,直至博弈结束的过程。

为了能够判断局部博弈树中各棋局节点的好坏,通常需要定义一个静态估价函数 $f(P)$, 用来估计棋局 P 的势态优劣。这个静态估价函数 $f(P)$ 可以定义为棋局 P 到我方胜利棋局的距离(如,胜利在望时: $f(P) \rightarrow +\infty$; 趋向败局时: $f(P) \rightarrow -\infty$; 势均力敌时: $f(P) = 0$)。 $f(P)$ 也可定义为棋局 P 到某个明显有利于我方棋局的距离,如吃掉对方一子,等等。

在生成一棵局部博弈树之后,可用静态估价函数 $f(P)$ 去度量该博弈树中各个端节点 P 的价值,而后采用倒推的办法,根据极小极大原则,自下而上地计算博弈树中各个非端节点的估价函数值。图 1 所示就是从 MAX 的立场出发构造的一棵深度为 2 的局部博弈树。树中各节点的估价函数值 $f(P)$ 的计算过程如下: 博弈树中各端节点给出的数字是根据

原定义的静态估价函数 $f(P)$ 计算所得,而其他非端节点的估价函数值则是应用倒推的办法,按极小极大原则求得的。

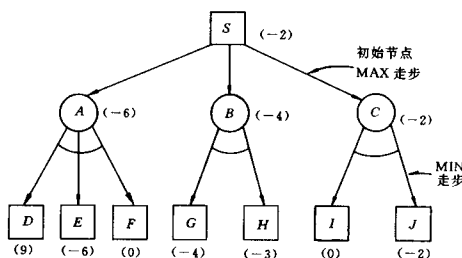


图1 深度为2的局部博弈树 $f(P)$ 求值过程

总之,有界深度极小极大博弈搜索过程分为两个阶段:第一阶段用宽度优先搜索方法生成规定深度的局部博弈树,然后计算树中各端节点的静态估价函数值;第二阶段根据极小极大原则自底向上逐级求树中各非端节点的倒推估价值,直至求至初始棋局节点 S 的 $f(S)$ 值为止。

选手可由 $f(S)$ 值选得相对最好的走步,搜索过程暂告结束。对手响应走步之后,再以当前棋局状态作为起始状态 S ,重复调用上述过程。

关于博弈树搜索的优化机制请参见 $\alpha\beta$ 剪枝过程。

参考文献

1. Nilsson N J 著. 人工智能原理. 石纯一等译. 北京: 科学出版社, 1984
2. 林尧瑞, 马少平. 人工智能导论. 北京: 清华大学出版社, 1989
3. 陆汝钤. 人工智能. 北京: 科学出版社, 1989
(康健初 杨莉)

bujianduan dianyuan

不间断电源 (uninterruptible power system, UPS)

当交流输入电源的变化(包括电压变化、频率变化及波形失真等)超出规定范围时,仍能正常地向负载输送能量的供电设备。不间断电源(UPS)可保证计算机工作时电源不间断。UPS分为旋转发电机式和静止变换式两大类,后来派生出动静结合式(不间断电池系统)。不间断电源主要由换能、储能和传输等部分构成。

不间断电源的种类及其结构

旋转发电机式 UPS 一种利用发电机来达到不间断供电目的的设备。它有三种:带飞轮的交流电

动机-交流发电机组,带电池组的直流电动机-交流发电机组和内燃机 UPS 系统。由于设备笨重现在已很少使用。

静止变换式 UPS 利用半导体器件、电池等组成的系统来实现不间断供电的设备。其基本结构如图1所示。

(1) 输入滤波器 主要的作用是抑制天电干扰、工业干扰及其他电磁干扰、浪涌等外来干扰,以保护 UPS 内部元器件和电路的安全。

(2) 整流器、充电器 将输入交流电压变成直流电压的装置。整流器一般采用可控硅整流电路,以达到稳压目的。

(3) 电池组 起储能作用。一般采用密封式免维护铅酸蓄电池,也有的采用碱性蓄电池,如铬镍电池等。

(4) 逆变器 将直流电压转换成交流电压的装置。

(5) 隔离变压器 将逆变器送来的交流电压转换成负载所需的值。

(6) 转换开关及其监控器 转换开关在其监控器的控制下,根据 UPS 的运行状态进行转换。

静止变换式 UPS 按工作模式可分为在线式、后备式及在线后备混合式,按其结构又可分为双变换式和线路交叉式。

(1) 后备双变换式 UPS(参见图1) 在电网正常供电时,UPS 的输出电压由工作旁路支路提供,只有当电网异常时,其转换开关才打向下方,输出电压才由电池组—逆变器—隔离变压器这一支路供给。在电网正常时,电网输入除一路向输出供电外,另一路经输入滤波器、整流器、充电器支路向电池组充电。后备式 UPS 的优点是效率高、可靠性高、结构简单;缺点是供电质量差,有切换时间,输出波形一般为方波或梯形波。

(2) 在线双变换式 UPS(参见图1) 无论市电正常与否,UPS 输出电压始终取自输入滤波器—整流器、充电器—逆变器—隔离变压器支路,这时转换开关的触点与下端(隔离变压器)相合。当逆变器故障或过载时,开关才打向上方,改由工作旁路供电。在线式 UPS 优点是供电质量高,市电断电时无切换时间(因多为静态开关),大都输出正弦波。缺点是效率低,结构复杂。

(3) 在线线路交叉式 UPS 基本结构见图2。线路交叉式与双变换式不同的是在旁路上加进了滤波器、电网电压调节器等装置。在电网正常时,两条

支路同时工作或交叉工作。当电网出现故障时,由下支路单独供电。线路交叉式具有高效率、结构简单、供电质量高和切换时间为零的优点。

(4) 在线后备混合式 UPS 结构原理见图 3。它由两条支路组成。当电网正常时,电网输入经过

整流器送逆变器,同时电网给电池组充电,如图中实线所示。当电网异常时,则由电池组通过直流变换器向逆变器供电,如图中虚线所示。由于采用了综合器,没有转换开关,切换时间为零。

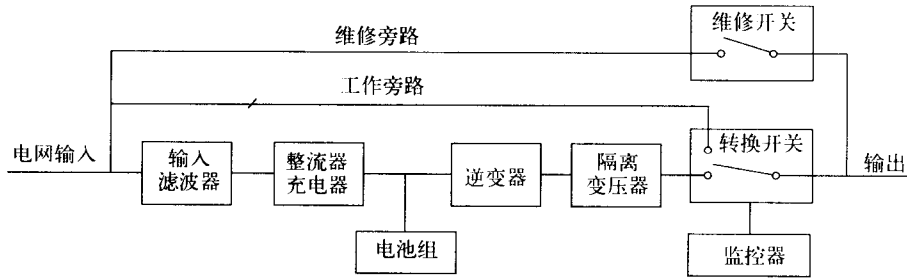


图1 静止变换式(双变换式)UPS基本结构图

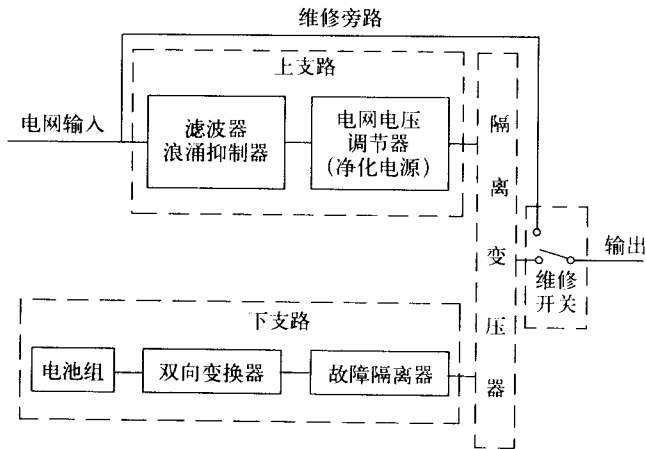


图2 线路交叉式UPS基本结构

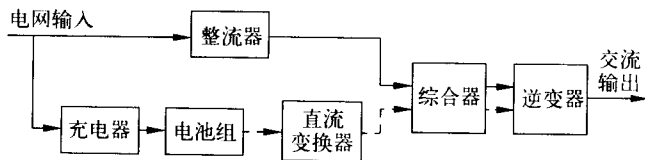


图3 在线后备混合式UPS结构原理

动静结合式 UPS 由一个静止变换式 UPS 和一个直流发电机组组成。当市电正常时,UPS 正常运转;一旦市电异常,UPS 由机内电池供电;如果市电短时间不能恢复,当电池放电到一定电平时,自动启动直流发电机为 UPS 机内电池充电,使电池的放电不间断地维持下去。

UPS 技术规格 UPS 的主要技术指标如下:

(1) 容量 指 UPS 的输出功率,其单位通常用 VA 表示。

(2) 输入电压范围 指 UPS 不用电池供电时,电网输入电压的最大值和最小值。

(3) 负载稳定度 当负载在指定范围变化时,

UPS 的输出电压变化的范围。

(4) 效率 UPS 输出功率与输入功率之比。

(5) 输出电压谐波失真 指 UPS 输出电压中高频分量所占比例。

(6) 音频噪声 UPS 工作时机内发出的可听见的声音。

(7) 切换时间 将负载从电网供电支路转换到电池供电支路或相反过程中,由于转换开关的作用,负载上会出现无电的一段时间,这段时间就是切换时间。

UPS 的应用与发展 广泛应用的 UPS 是静止变换式 UPS,它大量用于各种微型计算机供电系统中。由于微型计算机内采用的是开关电源,对输入波形要求不严格,正弦波、准方波和梯形波都能适应,因此常用后备双变换式 UPS。

对计算机联网系统,既要求供电质量高,又要求可靠性高,因此多采用在线式 UPS。为了提高可靠性,还要采取两台以上 UPS 并联、串联等冗余措施。先进的 UPS 内部采用微型计算机监控,通过 RS-232C 接口与外部通信,已成为计算机系统的一个有机部分。

除了在计算机领域外,UPS 在通信等领域也有广泛的应用。

参考文献

1. 中国电源学会交流稳定电源专业委员会编. 第二届 UPS 技术研讨会论文集. 1994
2. 张立, 赵永健. 现代电子电力技术. 北京: 科学出版社, 1992 (王其英 何春华)

buke panding wenti

不可判定问题 (undecidable problem) 没有求解算法的问题(参见可计算性理论)。集合的从属关系所对应的判定问题,若不是可判定的,则称为不可判定的。数学和计算机科学中提出的大量判定问题中,其绝大部分都是不可判定的。研究不可判定性,对一些具体的判定问题给出否定回答,常常具有重要意义。因为一旦某个判定问题被证明是不可判定的,则表明解决这个问题的能行办法是不存在的,不必再去寻找解决该问题的能行办法。

研究不可判定性的两个基本方法是直接方法和间接方法,直接方法是使用康托对角线化方法;间接方法就是将已知的不可判定问题归约到所研究的判定问题上,从而证明该判定问题比已知的不可判定问题更不可判定。利用归约不但可以间接地证明某

些判定问题是不可判定的,更重要的是归约关系在各种判定问题之间建立了不可判定程度的比较,依此确立了不可判定性的系统理论。

参考文献

1. Davis M D, Weyuker E J. Computability, Complexity and Languages, Fundamentals of Theoretical Computer Science. Academic Press Inc., 1983
2. 莫绍揆. 递归论. 北京: 科学出版社, 1987 (马绍汉 李大兴)

Buer daishu

布尔代数 (Boolean algebra) 设集合 B 含特定元素 0 和 1, 若 B 上的一元运算“ $-$ ”和二元运算“ $+$ ”与“ \cdot ”满足

- (1) $x+y=y+x$ 且 $x \cdot y=y \cdot x$ (交换律)
- (2) $(x+y)+z=x+(y+z)$ 且 $(x \cdot y) \cdot z=x \cdot (y \cdot z)$ (结合律)
- (3) $x+(y \cdot z)=(x+y) \cdot (x+z)$ 且 $x \cdot (y+z)=x \cdot y+x \cdot z$ (分配律)
- (4) $x+0=x$ 且 $x \cdot 1=x$ (零律)
- (5) $x+x=1$ 且 $x \cdot \bar{x}=0$

其中 $x, y, z \in B$, 则称 B 为一个布尔代数。例如在集合 $\{0, 1\}$ 上定义运算 $-$, $+$ 和 \cdot , 如下列表格所示:

$+$	0	1	\cdot	0	1	$-$	
0	0	1	0	0	0	0	1
1	1	1	1	0	1	1	0

即可获得一个布尔代数,通常称为逻辑代数。此外,命题代数和集合代数也都是常用的布尔代数。

布尔代数是英国数学家 G. Boole 在 1854 年首先提出来的,是为了把逻辑推理变为代数演算。到了 20 世纪 30 年代,随着计算机科学的飞速发展,布尔代数的近代理论才开始形成,并获得广泛的应用。M. H. Stone 在 1936 年证明了任意一个布尔代数都同构于一个集合代数。任意一个布尔代数也都同构于一个命题代数。此外还有原子布尔代数同构于该原子集合上的集合代数;完全原子布尔代数同构于该原子集合上的幂集代数。

一个布尔函数的最简式是指表示该函数的在某种意义上最简的多项式表达式,通常是指项数最少,或者文字个数最少,它与组合逻辑线路的最佳设计

密切相关。根据字母数的多少可分别采用奎因方法和卡诺图法求出其最简式。对于 2 值布尔函数, E. L. Post 在 1921 年证明了一个布尔函数系能生成所有布尔函数之充要条件为该系中含有非保 0、非保 1、非自对偶、非线性和非单调 5 类函数, 即著名的完备性定理。

设 $f(x_1, \dots, x_n)$ 是布尔函数, 一般 n 元布尔方程 $f(x_1, \dots, x_n) = 0$ 有解的充要条件为 $\prod_{\substack{\alpha_i=0,1 \\ i=1,\dots,n}} f(\alpha_1, \dots, \alpha_n) = 0$ 。采用逐次消元法可求出其特解、通解、再生通解和无冗余通解。

布尔代数中的 n 个元素 c_1, \dots, c_n 是正规的, 如果 $\sum_{i=1}^n c_i = 1$; 是直交的, 如果 $c_i c_j = 0, i \neq j, i, j = 1, \dots, n$; 是正交, 如果正规且直交。线性布尔方程 $\sum_{i=1}^n a_i x_i = 0$ 有正交解的充要条件为 $\prod_{i=1}^n a_i = 0$, 其正交参数解 (通解、再生通解) 为 $x_i = \sum_{j=1}^m b_{ij} t_j, i = 1, \dots, n$, 这里 b_{1j}, \dots, b_{nj} 正交, $j = 1, \dots, m; \sum_{j=1}^m b_{ij} \leq \bar{a}_i, i = 1, \dots, n$ 。

布尔矩阵有逆的充要条件为每一行正规 (直交) 且每一列直交 (正规), 其逆为其转置矩阵。矩阵 G 称为矩阵 A 的一个广义逆, 如果 $A = AGA$ 。根据不同的用途, 广义逆又有以下几种: 极小范数逆, $GA = (GA)^T$, 其中 $(GA)^T$ 是 GA 的转置矩阵; 最小二乘逆, $AG = (AG)^T$; 梯林-瓦格纳逆, $G = GAG$ 。

在布尔代数的定义中从左 (右) 边等式内将运算 + 与运算 · 互换, 0 与 1 互换后便可得右 (左) 边等式, 即具有对偶性, 从而布尔代数中的任意公式均具有对偶性。若在布尔代数中引入代数运算 $\oplus: x \oplus y = x \bar{y} + \bar{x} y$, 则布尔代数便成为一个环。这样, 布尔代数的理论与此特殊类型环的理论等价。由于布尔代数对逻辑规律进行了数学处理, 又在线路设计和自动化系统中有广泛应用, 故也称为逻辑代数、开关代数等。

参考文献

1. Rudeanu S. Boolean Functions and Equations. Amsterdam; North-Holland, 1974
2. Monk J D, Bonnet R. Handbook of Boolean Algebras. Vol ~ Vo3. Amsterdam; North-Holland, 1989
3. 金基恒著. 布尔矩阵理论及其应用. 何善培等译. 北京: 知识出版社, 1987 (罗铸楷)

butu jishu

布图技术 (layout technology) 按照工程要求把设计好的电路图转变为工艺图或掩模图的设计技术。布图设计主要包括布局和布线。布局是将元件的几何图形的布置优化; 布线是按优化规则将电气上相通的点连通。

印刷电路板的设计是将电子系统的电路图变为布线图的布图设计。印刷电路板上的元件几何图形及连线图统称为导电图形。其设计层数分为单层、双层及多层。其设计规则与所用的元件电气性能及加工工艺要求有关, 常用的有最小线宽, 线与线、线与焊盘、焊盘与焊盘的最小间隙, 最小走线长度, 焊盘的最小尺寸等。

集成电路的布图设计称为版图设计。集成电路的制造工艺复杂, 光刻是它的关键工艺之一。每次光刻都要用掩模版。因而集成电路的版图设计就成了集成电路设计和制造的关键步骤。集成电路的版图设计规则, 是设计人员在绘制各层几何图形时应遵守的一些限制。如最小宽度、最小间隔、最小延伸、最小重叠, 以及不同层的导体接触规则, 如接触孔形状、尺寸、CMOS 与 NMOS 规则等。

在 20 世纪 60 年代, 布图设计由手工进行。人工设计周期长, 效率低, 易出错。随着计算机图形显示技术的进步和发展, 在 70 年代后期开始应用计算机辅助设计技术, 出现版图半自动和自动设计系统。

计算机辅助版图设计是根据集成电路的性能要求, 先人工布局, 在坐标纸上绘制出以电路几何图形表示的总图的草图。然后用符号布局语言、版图图形编辑语言或人机交互版图设计系统, 将电路几何图形编排在版图上, 再把总图分层, 由与计算机相连的外部设备和专用设备 (如光学图形发生器、自动刻图机) 完成绘制总图、刻红模、制作初编版和精编版的工作。符号布局语言是用一些符号定义几何图形, 指定层属性、图形坐标位置值、图形尺寸值、图形重复间距及重复系数等。版图图形编辑语言除能直接描述版图的几何图形 (如矩形、多边形、连线、圆及圆弧), 还能提供图形编辑语句, 如插入、平移、镜像、旋转、删除、扩大、缩小、局部移动和重复等处理。人机交互版图设计为设计人员提供良好的设计环境。设计人员使用计算机辅助设计 (CAD) 系统提供的一些简明的选单, 能直接在图形显示器上生成和修改集成电路的布局设计, 包括画草图, 对单元设计进行压缩, 把单元图定义为符号, 重复安置, 确定芯片几何尺寸, 对设计差错及违反设计规则情况进

行检查等。

版图自动设计是指芯片上的几何图形由 EDA (参见**电子设计自动化**) 系统自动完成。设计工程师只要给出片子的逻辑描述,系统会自动地按照给定的逻辑描述产生芯片版图。版图自动设计方法分为门阵列、宏单元阵列、标准单元阵列、任意单元、多芯片等,分别说明如下:

(1) 门阵列设计方法 门阵列设计技术是利用预先制造好的“母片”来进行布图设计。母片中央部位以一定的间距成行成列地排列着基本单元电路。基本单元电路由 6~10 个晶体管组成,基本单元内的电阻、晶体管是预先制造好的。它的内部元件排列及其大小形状都相同。对基本单元内部元件进行不同的连线就可以构成如门、触发器、加法器等。母片的四周是输入输出单元,用作输入输出电平转换、输入保护、输出功率驱动等。芯片的最外围是焊点区。用门阵列实现一个逻辑系统需要逻辑变换、单元分配、自动布线等步骤。

(2) 宏单元阵列设计方法 把基本单元按列重复排列,基本单元之间有一个布线通道。功能简单的逻辑元件由一个基本单元构成,功能较复杂的逻辑元件由若干个基本单元相连接构成。这些逻辑元件称为宏单元。宏单元设计系统需要一个宏单元库。实现宏逻辑系统的步骤与门阵列相同。

(3) 标准单元阵列设计方法 把预先设计好的功能单元按列布局。功能单元可以是触发器或功能块。要求这些功能单元的版图设计成宽度相同,而高度可以不同。需要标准单元库存放这些单元的电子和逻辑特性以及几何尺寸。

(4) 任意单元设计方法 这是一种定制电路设计方法,又称积木块法。它利用预先设计好的功能单元的版图(门电路、触发器、寄存器、随机存储器、只读存储器等)进行电路的布图设计。要求单元版

图的形状是正交图形,但大小任意。单元引出线允许在单元的四边。在布图设计时,根据逻辑图的互连要求和单元的外形参数进行单元安置,留出布线通道区,通过合理的通道布线,实现电路互连。

(5) 多芯片设计方法 把多个芯片通过相互连线安置在同一个外壳封装中,这是近年来迅速发展的一种设计技术。

参考文献

1. Mead C A, Conway L A. 超大规模集成电路系统导论. 何诣译. 北京: 科学出版社, 1986
2. Muroga S. VLSI 系统设计. 茅于海, 刘玉玲译. 北京: 电子工业出版社, 1989
3. Pucknell D A, Eshraghian K. 超大规模集成电路设计基础. 王正华, 沈晓民, 党晓颖等译. 北京: 科学出版社, 1993 (潘雪增)

buxian xitong biao zhun

布线系统标准 (cabling system standard)

国内、外权威机构为规范布线行业行为而制定的技术标准。已制定的综合布线系统及其产品、线缆、测试标准主要有:

- (1) EIA/TIA 568 商用建筑物电信布线标准;
- (2) ISO/IEC 11801 国际标准;
- (3) EIA/TIA TSB-67 非屏蔽双绞线系统传输性能验收规范;
- (4) 欧洲标准: EN 5016、EN 50168、EN 50169, 是分别为水平配线电缆、跳线和终端连接电缆及其垂直电缆而制定的标准;

(5) 我国于 1995 年 3 月由中国工程建设标准化协会批准的《建筑与建筑群综合布线系统设计规范》。

(王晓东)

C

canshu guji

参数估计 (parameter estimation) 根据从母体抽得的样本,估计母体分布中包含的未知参数。通常一个母体的分布 $F(x, \theta)$ 依赖于一个或几个参数,它们是未知的。比如正态分布 $N(\mu, \sigma^2)$ 中的参数 μ 与 σ^2 , 泊松分布 $P(x, \lambda)$ 中的参数 λ 。现从母体 X 中抽取子样 X_1, X_2, \dots, X_n , 构造一个统计量 $T = T(X_1, \dots, X_n)$ 作为对 θ 的估计,通常记为 $\hat{\theta} = T(X_1, \dots, X_n)$, 这样得到的估计称为点估计。如果 $E\hat{\theta} = \theta$, 则称这个估计是无偏的。即使是一个无偏估计,也不能保证由 $\hat{\theta}$ 准确地估计出 θ 。可以证明,如果 $\hat{\theta}$ 是连续型随机变量,则 $P(\hat{\theta} = \theta) = 0$ 。用一个量估计 θ 是很难奏效的,于是就作出两个统计量 T_1, T_2 , 使得 $P(\omega: T_1 \leq \theta \leq T_2) = 1 - \alpha$, 其中 α 是事先给定的正数。称 $[T_1, T_2]$ 是 θ 的置信度为 $1 - \alpha$ 的区间估计, T_1 与 T_2 分别称为 θ 的置信下限和置信上限。区间估计的方法参见**假设检验**。点估计方法有两种:

矩估计 通过用子样矩估计母体矩的方法导出的参数估计称为矩估计。给定子样 X_1, X_2, \dots, X_n , 称 $\bar{X} \triangleq \frac{1}{n} \sum_{i=1}^n X_i$ 为子样均值, $s^2 \triangleq \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2$ 为子样方差, $\frac{1}{n} \sum_{i=1}^n X_i^r$ 为子样的 r 阶矩。用 \bar{X} 估计母体的数学期望 μ , 用 s^2 估计母体的方差都是矩估计。

极大似然估计 设母体 X 的密度函数为 $f(x; \theta_1, \dots, \theta_k)$, 其中 $\theta_1, \dots, \theta_k$ 为未知参数, X_1, \dots, X_n 是来自母体 X 的子样。于是 (X_1, \dots, X_n) 的联合分布密度为 $\prod_{i=1}^n f(x_i; \theta_1, \dots, \theta_k)$, 称

$$L(\theta_1, \dots, \theta_k) = \prod_{i=1}^n f(X_i; \theta_1, \dots, \theta_k)$$

为 $\theta_1, \dots, \theta_k$ 的似然函数。若有估计量 $\hat{\theta}_1, \dots, \hat{\theta}_k$ 使得

$$L(\hat{\theta}_1, \dots, \hat{\theta}_k) = \max \{L(\theta_1, \dots, \theta_k)\} \quad (1)$$

这里 \max 是对 $(\theta_1, \dots, \theta_k)$ 所有可能的取值而言, 则称 $\hat{\theta}_1, \dots, \hat{\theta}_k$ 分别为 $\theta_1, \dots, \theta_k$ 的最大似然估计。直观地说, (X_1, \dots, X_n) 落在 (x_1, \dots, x_n) 的邻域里的

概率是 $\prod_{i=1}^n f(x_i, \theta_1, \dots, \theta_k) dx_i$ 。显然, 不同的 $\theta_1, \dots, \theta_k$ 将会影响这个概率, 因为概率愈大的事件愈容易发生, 因此, 可将观察到的 (X_1, \dots, X_n) 取值为 (x_1, \dots, x_n) , 由此可认为, θ_j 应该是使 $\prod_{i=1}^n f(x_i, \theta_1, \dots, \theta_k)$ 达到了最大值。所以可用式(1)来确定 θ_i 的估计。

为了求得极大似然估计, 通常是求

$$\frac{\partial}{\partial \theta_j} \ln L(\theta_1, \dots, \theta_k) = 0, \quad j = 1, 2, \dots, k$$

的解 $\hat{\theta}_j = \hat{\theta}_j(X_1, \dots, X_n)$, $j = 1, 2, \dots, k$ 。

一个估计的好坏, 可从不同侧面提出一些标准, 如: 无偏性、最小方差无偏性、有效性、充分性等。

参考文献

成平, 陈希孺等. 参数估计. 上海: 上海科学技术出版社, 1985 (金治明)

canshu qumian

参数曲面 (parametric surface) 用参数表达式定义的曲面。它是**参数曲线**的表示形式在参数域上的推广, 参数曲线只有一个参数变量, 而参数曲面具有多个参数变量。在参数曲线中介绍的几种表示形式, 如 Ferguson、Bézier、B 样条、NURBS 等经扩展后均可用于表示参数曲面。

参数曲面在汽车、飞机、船舶、家用电器、建筑物、玩具等多种产品和工程的设计、制造以及动画、影视的制作中均广泛应用。

参数曲面可由一系列的曲面片拼合而成。

实际中常用矩形域上的参数曲面片, 它是由曲线边界包围的具有一定连续性的点集面片。若以两个参数的单值函数表示面片, 其表示式为

$$x = x(u, w)$$

$$y = y(u, w)$$

$$z = z(u, w)$$

$$p(u, w) = [x(u, w), y(u, w), z(u, w)],$$

$$u, w \in [0, 1]$$

参数曲面片的常用几何元素有以下几种:

(1) 角点 把 $u, w = 0$ 或 1 代入 $p(u, w)$, 得到 4 个角点 $p(0, 0), p(1, 0), p(0, 1)$ 和 $p(1, 1)$, 简记为 $p_{00}, p_{01}, p_{10}, p_{11}$ 。

(2) 边界线 矩形域曲面片的 4 条边界线是 $p(u, 0), p(u, 1), p(0, w), p(1, w)$, 简记为 $p_{u0}, p_{u1}, p_{0w}, p_{1w}$ 。

(3) 曲面片上一点 该点为 $p(u_i, w_j)$, 简记为 p_{ij} 。

(4) p_{ij} 点的切矢 在面片上一点 p_{ij} 处有 u 向切矢为 p_{ij}^u, w 向切矢为 p_{ij}^w 。

(5) p_{ij} 点的法矢 在 p_{ij} 处的法矢记为 $\mathbf{n}(u_i, w_j)$, 简记为 \mathbf{n}_{ij} ,

$$\mathbf{n}_{ij} = \frac{p_{ij}^u \times p_{ij}^w}{|p_{ij}^u \times p_{ij}^w|}$$

在矩形域上的常见的参数曲面形式有以下几种:

(1) 双三次参数曲面片 由 2 个三次参数变量 (u, w) 定义的曲面片。曲面片是参数曲面的基本单元。双三次参数曲面片是应用得最广泛的一种曲面片, 其代数形式是

$$P(u, w) = \sum_{i=0}^3 \sum_{j=0}^3 a_{ij} u^i w^j, \quad u, w \in [0, 1]$$

(2) Bézier 曲面 用 Bernstein 多项式及控制点网格定义的曲面。基于 Bézier 曲线, 可以给出 Bézier 曲面的表示式。设 $p_{ij} (i=0, 1, \dots, n; j=0, 1, \dots, m)$ 为 $(n+1) \times (m+1)$ 个空间点列, 则 $m \times n$ 次 Bézier 曲面定义为

$$S(u, w) = \sum_{i=0}^m \sum_{j=0}^n B_{i,m}(u) B_{j,n}(w) p_{ij}, \quad u, w \in [0, 1]$$

式中, $B_{i,m}(u) = C_m^i u^i (1-u)^{m-i}$, $B_{j,n}(w) = C_n^j w^j (1-w)^{n-j}$ 是 Bernstein 基函数。依次用线段连接点列 $p_{ij} (i=0, 1, \dots, n; j=0, 1, \dots, m)$ 中相邻两点所形成的空间网格, 称为控制点网格。

(3) B 样条曲面 用分段 B 样条多项式函数及控制点网格定义的曲面。基于 B 样条曲线, 可以得到 B 样条曲面的表示式。给定 $(n+1) \times (m+1)$ 个空间点列 $p_{ij} (i=0, 1, \dots, n; j=0, 1, \dots, m)$, 则 $S(u, w) = \sum_{i=0}^m \sum_{j=0}^n p_{ij} N_{i,k}(u) N_{j,l}(w)$, $u, w \in [0, 1]$ 定义了 $k \times l$ 次 B 样条曲面。式中 $N_{i,k}(u)$ 和 $N_{j,l}(w)$ 分别是 k 次和 l 次的 B 样条基函数, 由 p_{ij} 组成的空间网格成为控制点网格。

(4) 非均匀有理 B 样条曲面 用双参数非均匀有理 B 样条函数、控制点列及其与控制点列对应的权因子定义的面, 也称 NURBS 曲面。

由双参数变量分段有理 B 样条多项式定义的 NURBS 曲面是

$$S(u, v) = \frac{\sum_{i=0}^m \sum_{j=0}^n W_{ij} P_{ij} N_{i,p}(u) N_{j,q}(v)}{\sum_{i=0}^m \sum_{j=0}^n W_{ij} N_{i,p}(u) N_{j,q}(v)}, \quad u, v \in [0, 1]$$

式中, P_{ij} 是矩形域上控制点网格点列, W_{ij} 是相应控制点的权因子, $N_{i,p}(u)$ 和 $N_{j,q}(v)$ 分别是 p 次和 q 次的 B 样条基函数, 它们都是在节点矢量 $\mathbf{S}[s_0, s_1, \dots, s_{m+p+1}]$ 和 $\mathbf{T}[t_0, t_1, \dots, t_{n+q+1}]$ 上定义的。如果在非均匀参数轴上定义的节点矢量 \mathbf{S}, \mathbf{T} 具有下述形式:

$$\mathbf{S} = [0, 0, \dots, 0, \underbrace{s_{p+1}}_{(p+1) \uparrow}, \dots, s_{m+p+1}, \underbrace{1, 1, \dots, 1}_{(p+1) \uparrow}]$$

$$\mathbf{T} = [0, 0, \dots, 0, \underbrace{t_{q+1}}_{(q+1) \uparrow}, \dots, t_{n+q+1}, \underbrace{1, 1, \dots, 1}_{(q+1) \uparrow}]$$

则由 \mathbf{S}, \mathbf{T} 定义的曲面是非均匀有理 B 样条曲面, 简称 NURBS 曲面。

参考文献

1. Farin G. Curves and Surfaces for Computer Aided Geometric Design. Academic Press Inc., 1993
2. Mortenson M E. Geometric Modeling. John Wiley & Sons, 1985
3. 孙家广, 杨长贵. 计算机图形学 (新版). 北京: 清华大学出版社, 1995 (孙家广 冯结青)

canshu quxian

参数曲线 (parametric curve) 用参数表达式定义的曲线。如果参数用 t 表示, 则平面曲线上每一点笛卡儿坐标的参数式是:

$$x = x(t)$$

$$y = y(t)$$

该点坐标的矢量表示是:

$$\mathbf{P}(t) = [x(t), y(t)]$$

参数曲线的切矢量或导函数是:

$$\mathbf{P}'(t) = [x'(t), y'(t)]$$

不可能, 也无必要去研究 t 从 $-\infty$ 到 $+\infty$ 的整条曲线, 往往只对其中的某一部分感兴趣。通常将参数变量规格化, 使 t 在 $[0, 1]$ 闭区间内变化, 写成 $t \in [0, 1]$ 。只对此区间内的参数曲线进行研究。

常见的参数曲线形式有以下几种:

(1) Ferguson 曲线 用三次参数多项式定义的自由曲线。一条三次参数曲线的矢量形式是

$$\mathbf{P}(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0, \quad t \in [0, 1] \quad (1)$$

由式(1), 可以得到参数曲线的几何形式:

$$\mathbf{P}(t) = F_1 \mathbf{P}_0 + F_2 \mathbf{P}_1 + F_3 \mathbf{P}'_0 + F_4 \mathbf{P}'_1 \quad (2)$$

其中, P_0, P_1, P'_0, P'_1 为系数, 分别是曲线的两个端点 $P(0), P(1)$ 以及对应的切矢量 $P'(0), P'(1)$; $F = [F_1, F_2, F_3, F_4]$ 为调和函数

$$F_1 = 2t^3 - 3t^2 + 1, \quad F_2 = -2t^3 + 3t^2$$

$$F_3 = t^3 - 2t^2 + t, \quad F_4 = t^3 - t^2 \quad t \in [0, 1]$$

这种由端点及其切矢量定义的三次参数曲线称为 Ferguson 曲线或 Hermit 曲线。

(2) Bézier 曲线 用 Bernstein 多项式函数和控制点构造的参数曲线。Bézier 方法将函数逼近同几何表示结合起来。

给定 $n+1$ 个空间点列 $P_i (i=0, 1, \dots, n)$, 则 n 次 Bézier 曲线定义为 $C(t) = \sum_{i=0}^n P_i B_{i,n}(t), 0 \leq t \leq 1$, 式中 $B_{i,n}(t) = C_n^i t^i (1-t)^{n-i}$, 是 Bernstein 基函数, 也是曲线上各点坐标的调和函数。依次用线段连接点列 $P_i (i=0, 1, \dots, n)$ 中相邻两点所形成的折线集, 称为 Bézier 特征多边形, 与每一条曲线对应。曲线的起始点和终点与该多边形的起始点、终点相重合, 且多边形的第一条边和最后一条边表示曲线在起始点和终点处的切矢量方向。曲线的形状趋向于特征多边形的形状。

(3) B 样条曲线 用 B 样条函数构造的曲线。它除了保持了 Bézier 曲线的直观性和凸包性等优点外, 还可以进行局部修改, 且曲线更逼近特征多边形。曲线的阶次也与顶点数无关, 因而更方便灵活。

已知 $n+1$ 个控制点 $P_i (i=0, 1, \dots, n)$ 为特征多边形的顶点, k 阶 ($k-1$) 次 B 样条曲线的表达式为

$$C(u) = \sum_{i=0}^n P_i N_{i,k}(u)$$

其中, $N_{i,k}(u)$ 是 B 样条调和函数, 也称为 B 样条基函数, 按照递归公式可以定义为

$$N_{i,1}(u) = \begin{cases} 1 & \text{若 } t_i \leq u \leq t_{i+1} \\ 0 & \text{其他} \end{cases}$$

$$N_{i,k}(u) = \frac{(u-t_i)N_{i,k-1}(u)}{t_{i+k-1}-t_i} + \frac{(t_{i+k}-u)N_{i+1,k-1}(u)}{t_{i+k}-t_{i+1}}, \quad t_{k-1} \leq u \leq t_{n+1}$$

其中, t_i 是结点值, 结点是非减序列。当结点沿参数轴作均匀等距分布时, 则为均匀 B 样条函数; 反之, 则表示非均匀 B 样条函数。

(4) 非均匀有理 B 样条 (NURBS) 曲线 用非均匀有理 B 样条函数构造的曲线, 也称 NURBS 曲线。

用分段有理 B 样条基函数定义的 NURBS 曲

线是

$$C(u) = \frac{\sum_{i=0}^n W_i P_i N_{i,k}(u)}{\sum_{i=0}^n W_i N_{i,k}(u)} = \sum_{i=0}^n P_i R_{i,k}(u)$$

其中, P_i 是特征多边形顶点位置矢量, $N_{i,k}(u)$ 是 k 阶 B 样条基函数, W_i 是相应控制点 P_i 的权因子。结点矢量是

$$T = [\underbrace{\alpha, \alpha, \dots, \alpha}_{(k+1)\uparrow}, t_{k+1}, \dots, t_n, \underbrace{\beta, \beta, \dots, \beta}_{(k+1)\uparrow}]$$

用参数方程来表示曲线比用显式、隐式方程有更多的优越性: ①有更大的自由度来控制曲线的形状; ②可对曲线的参数方程直接进行几何变换 (如平移、比例、旋转等), 从而节省计算工作量; ③便于处理斜率为无穷大的问题, 不会因此而中断计算; ④参数方程中, 代数、几何相关和无关的变量是完全分离的, 而且变量个数不限, 这种变量分离的特点可以用数学公式去处理几何分量; ⑤采用规格化的参数变量 $t \in [0, 1]$, 可使其相应的几何分量是有界的, 而不必用另外的参数去定义其边界; ⑥易于用矢量和矩阵表示几何分量, 简化了计算。

参考文献

1. Farin G. Curves and Surfaces for Computer Aided Geometric Design. Academic Press Inc., 1993
2. Mortenson M E. Geometric Modeling. John Wiley & Sons, 1985
3. 孙家广, 杨长贵. 计算机图形学 (新版). 北京: 清华大学出版社, 1995 (孙家广 冯结青)

caozuo xitong

操作系统 (operating system) 管理硬件资源、控制程序运行、改善人机界面和为应用软件提供支持的一种系统软件。操作系统通常是最靠近硬件的一层软件, 它把硬件裸机改造成为功能更加完善的一台虚机器, 使得计算机系统的使用和管理更加方便, 计算机资源的利用效率更高, 上层的应用程序可以获得远较硬件所能提供的更多的功能上的支持。

形成与发展

第一代计算机速度慢, 外设少, 规模小, 程序员可直接控制程序的运行。随着计算机技术的发展出现了第二代计算机, 它的速度较高, 外设较多, 功能较强, 手工操作方式已不能适应。首先, 手工操作不能进行复杂的控制, 故不能满足功能较强的第二代

计算机的需要。其次,手工操作速度慢,会降低计算机的效率。例如,在第一代计算机上进行手工操作时算一个题花 1 h,其中控制操作费时 3 min,仅占 5%,这个算题用第二代计算机算可能只要 6 min,然而手工操作时间仍是 3 min,却占了 50%,这就难以承受。所以,设计一种软件来管理硬件资源和控制程序的运行已是不可缺少的了。操作系统就是在这种需求下,于 20 世纪 60 年代初期产生了。初期的操作系统又称为管理程序,其主要功能是控制输入输出设备,接收并执行操作员输入的命令和向操作员显示各种信息等。60 年代中期,操作系统的功能有了明显的发展,开始具有多道程序设计能力和分时操作功能。操作系统发展到这一阶段,它的作用和重要性已充分展示,成为计算机系统软件中最重要的一种。60 年代末 70 年代初操作系统的功能更加完善,出现了各种规模的操作系统,既有配置在大型计算机上的操作系统,如 MULTICS MVS 等,也有配置在中小型计算机上的操作系统,如 VM 370、UNIX 等。80 年代开始,微型计算机问世,又出现了 XENIX,CPIM,MS-DOS 等微型计算机操作系统。在这一阶段出现的操作系统中流行最广的要算 UNIX 操作系统了,UNIX 操作系统是美国贝尔实验室的 Ken Thompson 和 Demis Ritchie 于 1969 年研制出来的。它是一个分时操作系统,具有简易高效,易于理解和使用方便等特点,深受广大用户欢迎。他们在设计 UNIX 操作系统的同时还设计并实现了一种高级程序设计语言 C,UNIX 的绝大部分程序均用 C 语言写成。这就使得 UNIX 操作系统易于移植,可被各种不同型号的计算机采用,因此它被广泛地使用,成为应用最广泛的一种操作系统。

随着操作系统的发展,其功能愈来愈强,作用越来越大,对操作系统的研究也越来越受到重视。从 70 年代起开始对操作系统的功能设计和系统实现进行了广泛的研究。例如,人机通信,资源调度策略,同步机制,通信机制和容错的研究等。此外,对操作系统的实现,如操作系统的构件、结构设计方法、用高级程序设计语言书写操作系统等也开展了广泛的研究。

从 70 年代中期到 90 年代,又发展了若干种新型操作系统。70 年代中,为适应计算机网络发展的需要出现了网络操作系统。80 年代为了满足分布式系统和大规模并行系统的需要出现了分布式操作系统和并行操作系统。此外,还开展了智能操作系统的研究。

近年来随着开放系统的兴起和发展,适合开放系统的操作系统已成为一个重要的研究课题。开放操作系统的特性是符合国际标准,具有可扩充性、可移植性,1985 年成立的“IEEE 开放系统委员会”就把这种标准的操作系统命名为 POSIX,即计算机环境可移植操作系统。

分 类

迄今为止,操作系统大致有 7 种类型。①单用户操作系统,功能比较简单,每次只能支持一个用户程序运行。这种操作系统主要配置在功能较弱的微型计算机上。②批处理操作系统,能支持多个用户程序同时运行于一台计算机上,在批处理操作系统支持下,计算机系统可同时接受多个用户计算任务并成批地处理。③分时操作系统,它支持多个用户在各自的终端同时使用一台计算机而每个用户都感到好像自己各有一台独立的计算机。分时系统的用户可在各自的终端打入系统提供的命令来控制计算任务的执行,可从终端获得各种信息来了解计算任务的执行情况,可在任务执行中使用系统提供的各种资源如打印机、文件等。在分时操作系统支持下系统资源为多个用户共享,资源的利用率有了较大提高,系统的效率也有了较大改善。④实时操作系统,为实时系统配置的操作系统。它与其他操作系统的一个明显区别是它的实时特性,即要满足对时间的限制和要求。⑤网络操作系统,为计算机网络配置的操作系统。在网络操作系统支持下,网络中的各台计算机之间可以进行通信和共享资源。除了通信和资源共享外,还提供一些特殊的功能,如文件传输,将一个文件从一台计算机经网络传送至另一台计算机,远程作业录入,将一个计算任务送到他机去执行并将执行结果送回本机。⑥分布式操作系统,为分布式计算机系统配置的操作系统。这种操作系统在资源管理、进程通信和操作系统结构等方面都与经典的操作系统有较大的区别(参见分布式操作系统)。组成系统的各台计算机可以是同种型号的也可以是不同种型号的,相应的操作系统就是同构型操作系统和异构型操作系统。⑦并行操作系统,为大规模并行处理系统配置的操作系统。

基本内容

操作系统为系统资源的管理者提供硬件资源管理和程序控制功能,对使用者提供友好的人机界面,对应用程序则提供丰富的功能支持。

计算机资源可分为两大类:硬件资源和软件资

源。硬件资源指组成计算机的硬设备,如中央处理机、主存储器、磁带存储器、打印机、显示器、键盘输入设备等。软件资源主要指存于计算机中的各种数据和程序。

系统的硬件资源和软件资源都由操作系统根据用户需求按一定的策略分配和调度。操作系统的存储管理负责把内存单元分配给要执行的程序以便让它执行,在程序执行结束后将它占用的内存单元收回以便再使用。对于提供虚拟存储的计算机系统,操作系统还要与硬件配合做好页面调度工作,根据运行程序的要求分配页面,在执行中将页面调入和调出内存以及页面的回收工作等。

操作系统的处理器管理根据一定的策略将处理器交替地分配给系统内等待运行的程序。一道等待运行的程序只有在获得了处理器后它才能运行;一道程序在运行中若遇到某个事件,例如,它启动外部设备而暂时不能继续运行下去,处理器管理就要处理相应的事件,然后将处理器重新分配。常用的处理器分配和调度策略有优先数法和时间片法。采用优先数法时,进入计算机等待运行的程序都指派一个优先数,每次分配处理器时均按“先大后小”的原则,将处理器分配给等待运行中的优先数最大的那道程序。采用时间片法则是分配给每道程序一个时间片,若一道程序的连续运行时间超过指派的时间片时,就将它强行换下来排入等待运行程序队列的队尾而让队列的第一个等待运行的程序占用处理器。

操作系统的设备管理是分配和回收外部设备以及控制外部设备按用户程序的要求进行操作等。对非存储型外部设备,如打印机,可以直接作为一个设备分配给一个用户程序,在使用完毕后回收以便给另一个需要的用户使用。对于存储型的外部设备,如磁盘、磁带等,则是提供存储空间给诸用户存放各种程序和数据。

操作系统的文件管理是向用户提供创建文件、撤销文件、读写文件、打开和关闭文件等功能。有了文件管理,用户可以按文件名存取数据而无需知道这些数据存放在哪里。这种做法不仅便于用户使用而且还有利于用户共享公用数据。此外,由于文件建立时允许创建用户规定的使用权限,这就可以保证数据的安全性。

操作系统的人机交互功能是决定计算机系统“友善性”的一个重要因素。人机交互功能主要靠可输入输出的外部设备和相应的软件来完成。可供

人机交互使用的设备主要有:键盘显示、鼠标、各种模式识别设备等等。与这些设备相应的软件就是操作系统提供人机交互功能的部分。操作系统人机交互部分的主要作用是控制有关设备的运行和理解并执行通过人机交互设备传来的有关的各种命令和要求。早期的人机交互设施是键盘显示器,随着计算技术的发展,出现了语音输入设备、文字读入设备和图形、图像扫描输入设备,人机交互功能越来越强了。

操作系统的程序控制功能主要是控制用户程序的执行。一个用户将他要解决的问题用某一种程序设计语言编写成一个程序后就连同对程序执行的要求输入到计算机内,操作系统就根据要求控制这个用户程序的执行直到结束。操作系统控制用户程序的执行主要有以下一些内容:调入相应的编译程序将用某种程序设计语言编写的源程序编译成计算机可执行的目标程序,分配内存等资源将程序调入内存并启动,按用户指定的要求处理执行中出现的各种事件以及操作员联系请示有关意外事件的处理等。

结 构

操作系统是一种复杂程度高,生存周期长,正确性难保证的系统软件。所以,设计操作系统一般有以下三种要求:正确性、高效性、易维护性。采用良好的结构来实现操作系统是达到上述要求的关键技术。操作系统一般都采用内核加程序模块的结构。内核是操作系统的基本部分,它常驻内存,主要提供三方面的功能:中断处理、处理器调度、原语管理。内核一般都与机器有关,即位于不同计算机上的操作系统其内核往往是不同的,操作系统的其他部分都建立在内核之上。操作系统的结构设计方法有:模块接口法、层次分级法、核心扩充法等。

为了方便程序员编写应用程序,操作系统通常还以 API(应用编程接口)的形式提供许多功能丰富的子程序。这样,从程序员的角度看来,他所使用的机器不再是原始的硬件裸机,而是经过操作系统改造后的功能更加强大的一台虚机器。

参考文献

1. 孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989
2. Brown R L, Denning P J and Tichy W F. Advanced Operating System. IEEE Computer. Oct., 1984, 17(10)

(孙钟秀)

caozuo yuyi

操作语义 (operational semantics) 用解释执行程序的抽象机器定义语言的语义。

计算机语言的实现是在具体的计算机系统中按照语言的语义编制编译程序,将语言中各个成分翻译成计算机系统中相应的一组操作。语言在计算机系统的一种实现一旦完成,则对这个计算机系统而言,语言各个成分的含义也就完全确定了。因此语言的实现也可用来定义语言的语义。

操作语义的基本思想来源于程序设计语言的实现。1964年1月英国P. J. Landin使用“栈—环境—控制—存储”抽象机器(简称SECD机器),第一次系统地、严格地陈述了表达式的操作语义。

IBM公司的维也纳实验室在60年代研究程序设计语言PL/1的形式定义时,提出描述操作语义的一种元语言——维也纳定义语言(简称VDL)。1974年欧洲计算机制造商联合会和美国国家标准局正式建议使用VDL定义的PL/1语义作为PL/1的标准。

1980年前后,英国爱丁堡大学的计算机科学家提出结构式操作语义。它在一般的数学结构(不必是抽象机器)上用数学的归纳关系建立语义的解释系统。这种方法具有指称语义的结构式特征,并更多地略去了机器操作的细节。

程序设计语言中的算术表达式用于加工计算机系统中现存的数据,以形成新的数据。如表达式 $[(x_1 \times x_2) + 1]$ 表示将计算机系统中对应变量的存储单元数据相乘,再将乘积加1,形成一个新的数据。计算机系统中保存数据的存储区可以用数据向量来表示。 k 个存储单元保存的数据都是1时,可用 k 维向量 $(1, 1, \dots, 1)$ 表示。计算机系统中还要有保存程序的区域,以及保存加工过程中必要信息的工作区。就此可提出一种解释执行算术表达式求值的抽象机器的模型,这个机器的存储区分成三部分:栈区 st (用作工作区),环境区 s (保存数据向量等),控制区 c (保存程序)。整个存储区记作 (st, s, c) ,称为抽象机器的一个大状态。这个抽象机器具有识别符号、完成算术和逻辑运算、转储信息、实现大状态之间的转移等基本功能。

这个机器的大状态转移规则分为四类:

- (1) $(st, s, (e_1 \text{ op } e_2) / c) \Rightarrow (st, s, e_1 / e_2 \text{ op } / c)$
- (2) $(st, s, n / c) \Rightarrow (n / st, s, c)$
- (3) $(st, s, x_i / c) \Rightarrow (x_i / st, s, c)$
- (4) $(n / m / st, s, \text{op} / c) \Rightarrow (k / st, s, c) \quad (k = m$

$\text{op } n)$

第一类规则表示,当控制区中待执行的程序要求完成表达式 $(e_1 \text{ op } e_2)$ 的求值时,抽象机就转移自己的大状态,准备先求子表达式 e_1 和 e_2 的值,然后再按照相应的运算 $\text{op}(+, -, \times \text{或其他算子})$,求出整个表达式的值,符号“/”用于分割存放的信息。第二类规则表示,当求值的表达式是一个常量时,则其值就是抽象机中表示这个常量的相应的量(粗体用来区别语言中的符号和在抽象机中的相应表示),表达式的值暂存于栈区。第三类规则表示,当表达式是一个变量时,其值就是环境区中相应单元的当前值,即第 i 个变量 x_i 的值就是数据向量 s 的第 i 个分量 s_i 的值。第四类规则表示,当运算 op 的两个操作数已经求得,则可按照抽象机中的相应运算求出 op 作用于操作数的结果。

在这个抽象机中,表达式 $(x_1 \times x_2) + 1$ (在 x_1, x_2 值为2和3时)的求值是由下述大状态的转移序列完成的,转移符号 \Rightarrow 的上方标有实现这一转移依据的转移规则号,设 $s = (2, 3)$ 。

$$\begin{aligned}
 & (st, s, (x_1 \times x_2) + 1) / c \\
 & \stackrel{(1)}{\Rightarrow} (st, s, (x_1 \times x_2) / 1 + / c) \\
 & \stackrel{(1)}{\Rightarrow} (st, s, (x_1 / x_2) \times 1 + / c) \\
 & \stackrel{(3)}{\Rightarrow} (2 / st, s, x_2 / \times 1 + / c) \\
 & \stackrel{(3)}{\Rightarrow} (3 / 2 / st, s, \times / 1 + / c) \\
 & \stackrel{(4)}{\Rightarrow} (6 / st, s, 1 + / c) \\
 & \stackrel{(2)}{\Rightarrow} (1 / 6 / st, s, \text{eps}, +, / c) \\
 & \stackrel{(4)}{\Rightarrow} (7 / st, s, c)
 \end{aligned}$$

这个抽象机正确刻画出算术表达式求值的全过程,故可作为算术表达式的操作语义。

为定义赋值语句 $(x_i := e)$ 的操作语义,可在上述抽象机中添加如下转移规则:

$$\begin{aligned}
 (5) & (st, s, (x_i := e) / c) \Rightarrow (st, s, e / x_i := / c) \\
 (6) & (n / st, s, x_i := / c) \Rightarrow (st, s |_i^n, c)
 \end{aligned}$$

第五类规则表示,当抽象机执行 $(x_i := e)$ 时,先求出表达式 e 的值,然后再给 x_i 赋值。

第六类规则表示,对 x_i 赋以值 n ,相当于将数据向量 s 的第 i 个分量(即保存 x_i 当前值的存储单元)改为 n ,表示为 $s |_i^n$ 。

在定义程序设计语言的过程语句的操作语义时,由于不同的过程使用不同的环境,不同的栈区,不同的控制区,故引入外存储区,外存储区中保存有

当前未调用的所有过程的全部环境区、栈区和控制区,这种机器就是 Landin 的 SECD 抽象机。

人们希望语言的语义是独立于实现的,故操作语义中使用了抽象机器,但仍不可避免地涉及语言的实现,这是它的弱点。但同时又是它的优点,因为设计一个系统的过程可看作是一个求精的过程,将不可执行的功能描述,逐步求精为可执行的程序,而操作语义方法在求精过程的后期,可发挥其特有作用,可用于描述更多的执行细节。

参考文献

周巢尘. 形式语义学引论. 长沙: 湖南科技出版社, 1985
(周巢尘 李晓山)

ceshi gongju

测试工具 (testing tool) 用于发现软件中缺陷和错误的软件。又称程序测试工具。

软件是一种智力产品,随着软件规模和复杂性的不断增加,软件产品的可靠性和质量越来越难以保证。正像任何工业产品必须进行质量检验一样,如果软件产品不经过严格测试,其中隐藏的缺陷和错误便难以及时发现和排除,就可能造成重大损失,甚至酿成大祸。因此,未经测试的软件不能正式提交给用户使用,当然也不能作为产品推向市场。

软件测试要耗费大量人力物力,其费用一般要占项目总开发成本的 40% 以上。测试工具用于支持各项测试活动,尽可能发现程序中的缺陷、错误,确保软件质量。

软件测试的基本内容包括:

(1) 测试计划

在测试开始前,要针对被测对象制定包括测试内容、步骤、方法、标准、进度以及人员、资源等在内的测试计划,使整个测试工作能按规定要求进行。

(2) 软件度量

按照一定的度量理论和准则,对源程序的大小、结构复杂度 (McCabe, 1976 年) 和科学复杂度 (Halstead, 1977 年) 等进行度量,以辅助预测软件中最容易产生错误的部分,从而对制定测试计划和分配测试资源提供支持。

(3) 功能测试

又称黑箱测试。对软件功能规约或用户手册中所列的每项功能逐一进行测试。包括对正常和异常的输入(或操作)、出错处理、边界情况和极端情况等进行测试。

(4) 结构测试

又称白箱测试。对程序中的语句、分支、条件、基本路径和函数调用进行遍历测试。一般要求语句覆盖率达到 95% 以上,分支覆盖率达到 85% 以上。结构测试常用于单元(模块)测试。有关研究表明,结构测试可发现程序中大量常规方法难以发现的错误,查错有效率甚至可达 90% 以上。

(5) 回归测试

当软件修改以后,必须运行原有的全部测试用例重新测试,并验证测试结果。这样可确保修改后软件的正确性和质量。

软件测试的实际步骤通常为:

(1) **单元测试** 对单一模块(如函数)进行测试。

(2) **组合测试** 对若干模块构成的子系统或整个程序进行测试。

(3) **系统测试** 对完整的软件系统(甚至包括硬件)进行测试。

(4) **α 测试** 产品发布前开发单位的内部综合测试。

(5) **β 测试** 产品正式投入市场前,由有关用户试用测试。

软件测试工具可以分为以下几类:

(1) **数据获取工具** 获取各种测试数据的工具。

(2) **静态分析工具** 不执行测试用例,分析源代码的工具。

(3) **动态测试工具** 在执行测试用例过程中分析源代码的工具。

(4) **模拟工具** 模拟硬件环境及其他外部功能的工具。

(5) **测试管理工具** 测试计划、实施和进行测试控制的工具。

有的测试工具具有上述多种功能。

典型的测试工具列举如下:

静态分析程序 用来对程序的结构、模块调用与被调用关系、模块的控制结构以及变量、参数的定义与引用等进行静态分析,生成相应的信息和图示结果。

动态测试程序 能在被测程序中安装“探针”,使程序运行时自动记录执行情况,获得有关语句、分支条件和路径覆盖率的报告。

断言处理程序 可了解程序员规定的有关程序行为的各种断言在程序实际运行时是否相符。

截获-播放工具 在截获方式下,工具在程序的

特定点(通常是交互输入处)记录所有的信息流。当工具处于播放方式时,程序能自动输入在截获点处获取的数据运行。该工具特别适用于交互操作程序的回归测试。

测试数据生成程序 能自动分析被测程序,并按程序功能和测试要求辅助用户生成和选择测试数据。

测试结果分析程序 对测试输出信息进行分析、归类,产生各种测试结果报告和图表。

测试支持环境 支持软件测试过程。如安装、编辑和运行被测试程序,提供驱动模块和桩模块,输入测试数据,建立测试数据库,管理测试用例以及生成测试文档等。

死锁检测程序 专门检测并行、并发程序的进程通信同步的异常情况和可能出现的死锁情况。

测试计划辅助工具 提供测试计划模板,引导测试人员填写格式正确的测试计划。另外还可通过半形式化的语言来描述系统需求和测试目的,以辅助生成测试用例。

参考文献

1. Beizer B. Software Testing Techniques. 2nd ed. New York: Van Nostrand Reinhold, 1990
2. Sundermeier B. A Practical Software Testing Tool Adoption Strategy, CASE Trends. June, 1993

(金茂忠)

cengci shujuku

层次数据库(hierarchical database) 采用层次模型的数据数据库。

层次模型用树形结构表示各类实体及其间的联系。在树形结构中:

(1) 有且仅有一个结点没有双亲结点,这个结点称为根结点;

(2) 其他结点均有且仅有一个双亲结点。

在层次模型中,每个结点表示一个记录类型(简称记录型),结点之间的连线表示记录型间的联系,这种联系只能是一对多联系。每个记录型可包含若干个字段。记录型用来描述实体,字段用来描述实体的属性。

在层次数据库中,每个记录只有一个双亲记录(根记录除外),即从一个记录到其双亲记录的映射是惟一的,所以对于每一个记录,只需指出它的双亲记录就可以表示出层次模型的整体结构。层次数据库中,任何一个给定的记录值只有按其路径查看时,

才能显出它的全部意义,没有一个子女记录值能够脱离双亲记录值而独立存在。

在层次模型中,具有同一双亲的子女结点之间互称兄弟结点,没有子女结点的结点称为叶结点。

图1是一个层次模型的例子:其中学校为根结点;系和研究所是学校的子女结点,它们互为兄弟结点;教研室和班级是系的子女结点;教研室、班级和研究所是叶结点。

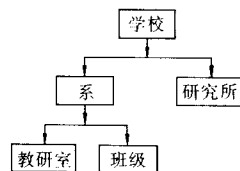


图1 一个层次模型

储存层次数据库不仅要储存数据本身,还要反映出数据之间的层次联系,实现方法有两种:

(1) **邻接法** 按照树的某种遍历的顺序(例如深度优先、广度优先)把所有记录值依次邻接存放,即通过物理空间的位置相邻来实现层次顺序。

(2) **链接法** 用指引元来实现数据之间的层次联系。每个记录设两类指引元,分别指向最左边的一个子女(每个记录型对应一个)和最近的兄弟。这种链接方法称为子女—兄弟链接法。另一种链接方法是按树的先根(次序)遍历的顺序链接各记录值,称为层次序列链接法。

信息管理系统IMS是最早的层次数据库管理系统,由IBM公司研制,先后推出了多个版本。1968年的IMS-1支持HSAM和HISAM存储结构;1971年的IMS-2在IMS-1的基础上又增加了HDAM、HIDAM储存结构和逻辑数据库;1974年的IMS/VS又增加了批处理检查点、重新启动、并发操作、辅助索引等功能。

现实世界中许多实体之间的联系本来就呈现出一种自然的层次关系,如行政机构、家族关系等。用层次模型对具有一对多层次关系的部门进行描述非常自然、直观、容易理解。这是层次数据库的突出优点。

层次数据库也存在一些弱点。在层次数据库中,双亲记录与子女记录之间只能反映一对多的联系,而现实世界中很多联系是非层次性的,如多对多联系、一个结点具有多个双亲等。层次模型表示这类联系的方法很笨拙,只能通过引入冗余数据(易

产生不一致性)或创建非自然的数据组织(引入虚拟结点)来解决。另外,层次数据库用存取路径来表示数据之间的联系,用户对数据的存取必须按照明确定义的存取路径进行,必须清楚地了解数据在数据库中的当前位置,加重了用户的负担;对数据的操作是一次一个记录的存取方式,程序和数据虽有较高的物理独立性,但逻辑独立性不高。

层次数据库系统在20世纪70年代与80年代初非常流行,在数据库系统产品中占主导地位,虽然近年来逐渐被关系数据库系统取代,但在美国、加拿大等国家,由于历史原因,目前层次数据库的使用仍很多。

参考文献

1. 萨师煊,王珊.数据库系统概论.第二版.北京:高等教育出版社,1991
2. Date C J. An Introduction to Database System. 6th Edition. Reading: Addison Wesley Publishing Company, 1995 (王珊)

chazhi

插值(interpolation) 关于如何寻找函数 φ 来近似地代替任意函数 f ,并使得 φ 和 f 在若干点上具有相同值的方法与过程。

插值的作法是:在事先选定的一个由简单函数所构成的含 $n+1$ 个参数 c_0, c_1, \dots, c_n 的函数类 $\varphi(c_0, c_1, \dots, c_n)$ 中求出满足条件

$$p(x_i) = f(x_i) \quad i = 0, 1, \dots, n \quad (1)$$

的函数 $p(x)$,并以 $p(\bar{x})$ 作为 $f(\bar{x})$ 的估值。此处,函数 $f(x)$ 称为被插值函数; x_0, x_1, \dots, x_n 称为插值结点; $\varphi(c_0, c_1, \dots, c_n)$ 称为插值函数类;式(1)称为插值条件。 $\varphi(c_0, c_1, \dots, c_n)$ 中满足插值条件(1)的函数 p 称为插值函数,误差 $R(x) = f(x) - p(x)$ 称为插值余项,它标志着插值精度。当估值点 \bar{x} 属于包含结点 x_0, x_1, \dots, x_n 的最小闭区间时,称相应的插值为内插,否则称为外插。

插值是观测数据处理和函数制表所常用的方法,又是数值积分、数值微分、微分方程数值解、非线性方程求根、曲线或曲面拟合的依据。

常用以下插值方法。

多项式插值 插值函数类取作代数多项式类的情形,是最常用的一种插值。此时对 $[a, b]$ 上的任何实值函数 f 都有惟一的次数不超过 n 的多项式函数 p 满足插值条件(1), p 称为 f 的插值多项式函数。当 f 在 $[a, b]$ 上 $n+1$ 次可微时,插值余项为

$$R(x) = f(x) - p(x)$$

$$= \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{i=0}^n (x - x_i)$$

其中 ξ 是包含 x, x_0, x_1, \dots, x_n 的最小闭区间中的某一点。适当选择结点 x_0, x_1, \dots, x_n ,可得到较好的插值效果。特别,当结点 x_k 取作切比雪夫多项式的零点,即

$$x_k = \cos \frac{2k+1}{2(n+1)}\pi \quad k = 0, 1, \dots, n$$

时,在一定的意义下,是插值结点的最佳分布。

常见的两种多项式插值公式如下:

(1) 拉格朗日插值公式

$$p(x) = \sum_{i=0}^n f(x_i) l_i(x)$$

其中

$$l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x - x_j}{x_i - x_j} \quad i = 0, 1, \dots, n \quad (2)$$

称为拉格朗日插值公式的基函数。它们具有性质

$$l_i(x_j) = \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases}$$

(2) 牛顿插值公式

$$p(x) = f(x_0) + f(x_0, x_1)(x - x_0) + \dots$$

$$+ f(x_0, x_1, \dots, x_n)(x - x_0)(x - x_1) \dots (x - x_{n-1})$$

式中, $f(x_0, x_1, \dots, x_k)$ 为函数 f 在点 x_0, x_1, \dots, x_k 上的 k 阶差商。各阶差商由下列递推公式确定:

$$f(x_0, x_1) = \frac{f(x_0) - f(x_1)}{x_0 - x_1}$$

$$f(x_0, x_1, x_2) = \frac{f(x_0, x_1) - f(x_1, x_2)}{x_0 - x_2}$$

.....

$$f(x_0, x_1, \dots, x_n) = \frac{f(x_0, x_1, \dots, x_{n-1}) - f(x_1, x_2, \dots, x_n)}{x_0 - x_n}$$

拉格朗日插值公式和牛顿插值公式是同一插值多项式 $p(x)$ 的不同表现形式,因此它们有相同的插值余项。但在实际计算中,后者较为方便,若要增加新的插值结点,只需相应地添加新的项即可。

埃尔米特插值 带有导数值的插值问题。即插值条件为

$$H(x_i) = f(x_i), H'(x_i) = f'(x_i)$$

$$i = 0, 1, \dots, n \quad (3)$$

在所有次数不超过 $2n+1$ 的多项式中,满足插值条件(3)的多项式是存在且惟一的,并可表示为

$$H(x) = \sum_{i=0}^n [f(x_i) [1 - 2(x - x_i) l_i^1(x_i)] l_i^2(x) + f'(x_i) l_i^1(x)]$$

$$+ f'(x_i)(x - x_i)l_i^2(x) \} \quad (4)$$

式中, $l_i(x)$ 为拉格朗日插值基函数。 H 称为函数 f 的埃尔米特插值多项式函数。当 f 在 $[a, b]$ 上是 $2n + 2$ 次可微时, 插值余项为

$$R(x) = f(x) - H(x)$$

$$= \frac{f^{(2n+2)}(\xi)}{(2n+2)!} (x - x_0)^2 (x - x_1)^2 \cdots (x - x_n)^2$$

式中 ξ 是包含 x, x_0, x_1, \dots, x_n 的最小闭区间中的某一点。由于埃尔米特插值多项式在结点处与被插值函数取值相同, 且变化率也相同, 因此, 它通常比拉格朗日插值多项式能更好地近似被插函数。式 (4) 是一种最基本、最重要的埃尔米特插值多项式。此外, 在插值结点上, 作为插值条件, 还可以给出逐次高阶微商值, 并且在每个结点上的插值条件个数也可以是互不相同的。

分段插值 在被插值函数不够光滑或插值结点选择不当时, 高次插值多项式常常在被插值函数附近激烈地摆动, 不能逼近被插函数。其次, 高次插值多项式常常将插值条件的数据中含有的误差过分地放大和扩散。因此, 实际上很少采用高次多项式插值, 而是先将全区间分成许多小区间, 然后在每个小区间上采用低次插值 (一次, 二次或三次插值)。这样的方法称为分段插值法。实践表明, 用分段低次插值多项式逼近被插值函数往往比在全区间上用高次插值多项式逼近效果更好。

样条插值 它是一种在相邻插值曲线段的连接处具有一定光滑度的分段插值。最常用的是三次多项式样条插值: 给定结点 $x_0 < x_1 < \dots < x_n$ 和相应的函数值 $f(x_0), f(x_1), \dots, f(x_n)$, 取内结点 x_1, x_2, \dots, x_{n-1} 作为分段点, 寻求一个插值函数 $S(x)$, 它在每个小区间 $[x_{i-1}, x_i]$ ($i = 1, 2, \dots, n$) 上分别都是三次多项式, 在结点处满足插值条件

$$S(x_i) = f(x_i) \quad i = 0, 1, \dots, n$$

并在每个分段点处满足直到二阶微商的连续性条件

$$S(x_i - 0) = S(x_i + 0)$$

$$S'(x_i - 0) = S'(x_i + 0)$$

$$S''(x_i - 0) = S''(x_i + 0)$$

$$i = 1, 2, \dots, n-1$$

这里共有 $4n - 2$ 个条件, 而分段三次多项式 $S(x)$ 在每个小区间上均含有 4 个系数, 共计 $4n$ 个待定系数。因此, 在端点 x_0 和 x_n 处分别给定一个边界条件之后, 插值问题有惟一解。更一般地, 若插值函数 $S(x)$ 为分段 k 次多项式, 在诸结点上取给定值, 并

且在各分段点处有直到 $k - 1$ 阶的连续微商, 则 $S(x)$ 称为 k 次多项式样条插值。

三角插值 插值函数类为三角多项式函数类的情形。当被插函数 f 是以 2π 为周期的函数时, 通常用 n 阶三角多项式

$$T(x) = a_0 + \sum_{k=1}^n (a_k \cos kx + b_k \sin kx)$$

作为插值函数类。对于任取的 $2n + 1$ 个不同的实数 x_0, x_1, \dots, x_{2n} , 只要它们两两之差都不是 2π 的整数倍, 则满足插值条件

$$T(x_i) = f(x_i) \quad i = 0, 1, \dots, 2n$$

的阶数不超过 n 的三角多项式 $T(x)$ 存在并且惟一。同拉格朗日插值公式的结构相似。 $T(x)$ 可表示为

$$T(x) = \sum_{i=0}^{2n} f(x_i) \left(\prod_{\substack{j=0 \\ j \neq i}}^{2n} \frac{\sin \frac{x - x_j}{2}}{\sin \frac{x_i - x_j}{2}} \right)$$

有理插值 插值函数类取为有理函数类的情形。当被插函数具有极点或其他奇点时, 用有理插值往往很有效。假定已知被插函数 $f(x)$ 在 $m + n + 1$ 个互不相同的点 x_0, x_1, \dots, x_{m+n} 处的值为 $f(x_0), f(x_1), \dots, f(x_{m+n})$, 有理插值就是求一个形如

$$R_{m,n}(x) = \frac{a_0 + a_1x + \dots + a_mx^m}{b_0 + b_1x + \dots + b_nx^n} = \frac{P_m(x)}{Q_n(x)}$$

的有理分式函数, 使其满足插值条件

$$R_{m,n}(x_i) = f(x_i) \quad i = 0, 1, \dots, m+n$$

并以 $R_{m,n}(x)$ 作为 $f(x)$ 的近似值。当 $R_{m,n}(x)$ 的分子和分母有公因式时, 可约掉公因式, 将所得的既约分式和原来的分式看作同一分式。满足上述插值条件的有理分式 $R_{m,n}(x)$ 若存在, 则是惟一的。虽然 $R_{m,n}(x)$ 有时可能不存在, 但很多情况下, $R_{m,n}(x)$ 还是存在的, 其系数 a_i 和 b_i 可通过解由插值条件导出的线性方程组

$$Q_n(x_i)f(x_i) = P_m(x_i) \quad (i = 0, 1, \dots, m+n)$$

得出。

参考文献

Гончаров В Л 著. 函数插补与逼近理论. 路见可译. 北京: 科学出版社, 1958 (沈毅)

chaxun chuli

查询处理 (query processing) 数据库管理系统执行查询语言的数据操纵语句的过程。

查询处理大致可以分为以下 4 个阶段。①语法分析, 对给定的数据操纵语句进行分析, 报告语句中

可能出现的语法错误。如果没有语法错误,就转入下步处理。②语义检查,根据有关的模式定义、安全性规则及完整性约束对查询进行语义检查,报告有关错误。如果通过检查,就将原始语句转换成适合于后续处理的内部表示形式。③查询优化,对查询的各种可能的执行方案进行成本评估,选择最低廉的方案作为执行计划。执行计划是查询优化阶段的输出,由实现原始操作的可执行的内部代码组成。④执行,系统执行优化阶段产生的执行计划,回送查询结果。

参考文献

1. Date C J. A Guide to INGRES. Reading: Addison-Wesley Publishing Company, 1987

2. Gardarin G, Valduriez P. Relational Databases and Knowledge Bases. Reading: Addison-Wesley Publishing Company, 1989
(周傲英)

chaxun youhua

查询优化 (query optimization) 在关系数据库管理系统(RDBMS)中,对给定的查询选择高效的执行计划的方法与过程。所谓执行计划亦称查询树,它由一系列内部操作符组成。这些操作符按一定的运算关系构成查询的一个执行方案。对一个给定的查询,有许多可能的执行计划,为了达到用户可接受的性能 RDBMS 必须对各种执行方案按照一定的优化方法,选择一种高效的方案作为执行计划。高效的执行计划有两方面的衡量标准:一是使中间结果最小化;另一是采用尽可能好的操作算法。

传统的查询优化,按优化层次或优化内容分为代数优化和非代数优化,亦称逻辑优化和物理优化。前者主要是依据关系代数的等价变换进行逻辑变换,以求得优化。后者则主要是根据存储路径和排序来优化数据存取。

统计信息是查询优化的基础,通常包括一个关系中的元组数、关系中元组的长度、关系中在特定属性值上的分布情况、索引等。根据这些统计信息,我们可以估算各种运算结果的大小,以及运算的执行代价。例如,在有几个索引可用的情况下,利用统计信息可以有效地判定最佳的存取路径。

查询优化的基本方法有基于代价的方法和启发式方法。

基于代价的方法首先要生成所有可能的候选计划,通过统计信息和存取路径确定每一计划中的各个操作的代价和综合代价,最后选择代价最小的一

个。基于代价的方法的最大问题是优化本身的代价较大。

为降低优化代价,许多系统使用了启发式规则来减少候选计划。一些早期的系统甚至只采用启发式方法进行优化。由于启发式方法基于假定因素太多,虽然优化的代价较小但优化的效果较差,目前已很少单独使用。

现在,实际 RDBMS 的优化器都将两种方法结合起来使用,即利用启发式方法尽量减少候选计划,利用基于代价的方法准确地确定执行计划。只是各个系统在具体做法上稍有不同而已。

除以上两种基本的优化方法外,还有语法优化和语义优化方法。

语法优化就是系统依据用户给出的查询语法确定其执行计划。语法优化依赖于用户对数据库模式和表中数据存储分布情况的充分了解。它对相对静态环境下的数据访问是极其有效的。但是对经常访问动态变化数据的嵌入 SQL 应用,语法优化就显得颇不方便。

语义优化则建立在系统对数据库模式的理解上。对用户提交的查询,优化器可以利用系统掌握的知识来简化查询计划,或省略结果为空的查询计算。不过语义优化在目前实际系统尚未实现。

现时的应用环境日益复杂,高效率的应用需求越来越高,并行计算环境日益普及,因此适应这些新的应用环境和需求的查询优化方法不断出现,从而弥补原有方法的不足。一个重要的改进方法是流水线处理策略。

参考文献

1. Silberschatz A, Korth Henry F, Sudarsham S. Database System Concepts (3rd edition). McGraw-Hill, 1999

2. 萨师煊,王珊. 数据库系统概论. 北京:高等教育出版社,2000
(孟小峰)

chanpin shuju guanli

产品数据管理 (product data management, PDM) 一种以产品为核心,实现对产品相关的数据、过程、资源一体化集成管理的软件技术。

从总体上看,产品数据管理(PDM)提供给用户的主要功能有以下方面:

(1) 数据和文档管理 其管理的对象是各种文档(包括设计文件、由其他应用程序生成的文件等)、产品数据以及相应的属性和版本方面的信息。

(2) 过程和工作流管理 控制 PDM 操作人员产生和修改数据的方法,为企业建立流程和有效执行任务的框架,以提高企业内部的协同和并行工作水平。

(3) 产品结构与管理 提供在产品的整个生存周期建立和管理产品定义和结构的能力。

(4) 零部件分类库管理 可以更好地管理类型繁杂、数量巨大的零件数据,充分利用标准件、通用件等现有信息。

(5) 计划与项目管理 是关于企业的项目信息的管理模块,涉及项目任务和期限的指派、项目资源的分配、人员组织结构、人员角色分类、用户信息库、瓶颈分析以及触发提醒等内容。

产品数据管理(PDM)的体系结构由 4 层组成:

①数据库核心层 是系统的支撑层,提供数据管理最基本的功能;②对象库核心层 是信息管理对象的核心,提供描述产品数据动态变化的数学模型,并具有对 PDM 各类对象的管理功能;③应用系统层 也称功能层,提供系统功能的管理界面和应用界面;④应用软件层 也称用户层,包括应用软件模块(如用户化工具)和应用程序开发界面。

产品数据管理(PDM)的发展可以分为以下 3 个阶段:①配合计算机辅助设计(CAD)工具的 PDM 系统 早期的 PDM 产品诞生于 20 世纪 80 年代初。其目标主要是解决大量产品数据的储存和管理问题,提供“电子绘图仓库”的维护功能。②专业 PDM 产品 PDM 产品中出现了许多新功能,同时软件的集成能力和开放程度也有较大的提高,PDM 开始成为一种产业。出现了许多专业开发、销售和实施 PDM 的公司。③PDM 的标准化阶段 1997 年 2 月,(分布网络环境)对象管理组(OMG)组织公布了 PDM Enable 标准草案。PDM Enable 是 PDM 领域的第一个国际标准,标志着 PDM 技术在标准化方面迈出了新的一步。

从目前的趋势看,PDM 将会在以下几方面有较大发展:①网络技术在 PDM 系统中的应用越来越深入。基于网络平台和 Java 语言开发结构灵活、界面友好的 PDM 系统已成为一种趋势。②PDM 与物流需求计划(MRP)(参见企业资源规划)的功能渗透。二者之间通过互相集成,构成完整的企业信息系统。③基于万维网(Web)技术的 PDM 系统。PDM 系统架构在 Internet、内联网和外联网之上,以满足企业电子商务的需求。④PDM 向整体集成解决方案发展。其趋势是从强调单个特征和功能的

“技术工具包”发展到强调“问题解决方案”的系统和服务。⑤PDM 向全企业级方向发展。对于一些大型企业来说,要求 PDM 系统的功能覆盖企业全部经营范围。这时,由核心 PDM 程序、局部数据管理器以及集成工具共同构成的企业多级 PDM 系统将尝试新的 PDM 解决方案。

参考文献

1. 李善平,刘乃若,郭鸣. 产品数据标准与 PDM. 北京:清华大学出版社,2002

2. 高奇微,莫欣农. 产品数据管理(PDM)及其实施. 北京:机械工业出版社,1998 (汤文成)

chanpin shuju jiaohuan biao zhun

产品数据交换标准(product data exchange specification, PDES)

美国标准和技术协会(NIST)主持制定的产品数据交换标准。NIST 于 1989 年 4 月公布了 PDES 1.0 标准。其草案被国际标准化组织(ISO)接受后已作为 STEP 公布。1983 年 12 月,国际标准化组织(ISO)成立了 184 技术委员会(TC184,工业自动化系统),下设第四分委员会(SC4)的领域是产品数据交换,其制定的标准被称为产品数据表达与交换的国际标准,一般称其为 STEP 标准,ISO 的正式代号为 ISO 10303。

STEP 标准涵盖一个中性格式的计算机化的产品模型,在产品的整个生命周期内保证产品信息的完整性。它基于图形交换规范 IGES、美国的 PDES 标准、德国汽车 CAD 数据交换规范 VDA-FS、法国 CAD 数据交换规范 SET、欧洲 ESPRIT 的 CAD-I 和产品定义数据接口 PDDI。STEP 标准初次发布于 1994 年,数字化产品数据必须包含整个产品生存周期内的所有信息,从设计到分析、制造、质量控制、检查和产品服务功能。所以 STEP 标准包含几何、拓扑、公差、关系、属性、装配、结构等方面的信息。

STEP 标准不仅是一项标准,而是一组标准的总称。它是关于产品数据表示和交换的国际标准。STEP 不但适合于物理文件交换,而且还是实现和共享产品数据库及产品数据长期存档的基础。STEP 的特点反映在以下方面:①便携和简便 STEP 提供了一种中性机制来描述产品数据字典。并可以通过扩展和完善来包括新的数据类型,而不用重新设计转换程序。②兼容性 STEP 能够使各类用户通过不同的应用系统同时访问数据库、实现数据共享。③寿命周期长 STEP 提供描述产品生存周期内产品数据的机制。产品是与产品设计、分析、支持等过

程有关的函数。④可扩展性 产品数据的描述是明确的,能表示不同的数据类型,具有可扩展性。

STEP 主要由应用层、逻辑层和物理层 3 层结构组成。它包含了完整的产品数据模型,如几何形状、拓扑信息、形位公差、表面粗糙度、材料特性、工艺特性、设计特性、功能特性、装配结构、有限元分析等。

STEP 集成工具为制造业实施其战略目标开拓了一条途径。这些战略目标包括:并行工程、连续获取生命周期支持 CALS、虚拟企业及其协同工作、用户需求的敏捷反应能力、缩短产品开发周期、极大地减小产品生命周期成本、保证产品数据的完整性和一致性等。因此,基于 STEP 标准的产品数据的定义、建模、传输与交换的软件开发和产业化应用具有很大的实际意义。

参考文献

红军,张怀存. STEP 标准及其企业信息化. 机械加工与自动化,2003(5) (汤文成)

chanshengshi

产生式 (production) 定义语言文法的规则。其一般形式是 $\alpha \rightarrow \beta$, 其中 α, β 为由文法的非终极符和终极符组成的串,且可为空。它说明箭头左边的串能用箭头右边的串来替换。

语言的所有文法规则构成一产生式集合,使用这些规则,可产生出所有由终极符构成的串,这种字符串称为(该语言的)句子。

考虑下面的产生式集合:

- (1) $S \rightarrow aSBC$
- (2) $S \rightarrow aBC$
- (3) $CB \rightarrow BC$
- (4) $aB \rightarrow ab$
- (5) $bB \rightarrow bb$
- (6) $bC \rightarrow bc$
- (7) $cC \rightarrow cc$

(这里,大写字母 S, B, C 为非终极符,小写字母 a, b, c 为终极符),则此产生式集定义的语言等于集合 $\{a^n b^n c^n | n \geq 1\}$ 。因为对每一个 $n \geq 1$,反复使用产生式(1) $n-1$ 次得到串 $a^{n-1} S(BC)^{n-1}$ 。然后,对此结果使用产生式(2)一次得到串 $a^n (BC)^n$ 。再重复使用产生式(3) n 次,将所有的 B 都先于所有的 C ,即得串 $a^n B^n C^n$ 。其次,使用产生式(4)一次得 $a^n b B^{n-1} C^n$ 。再使用产生式(5) $n-1$ 次得 $a^n b^n C^n$ 。最后使用产生式(6)一次继而使用产生式(7) $n-1$ 次,得到终极符串 $a^n b^n c^n$ 。

一般说来,要确定产生式集产生的语言是极其困难的。但对上面的产生式集,可以证明它恰恰产生那个集合。

现在考虑稍为复杂一点的产生式,用它可定义上下文有关语言:

$$S_1 S S_2 \rightarrow S_1 T S_2$$

它表明:如果串 S 出现在上下文 S_1 与 S_2 中(即在串 S_1 与 S_2 之间),则 S 可以由串 T 来替换。因此,产生式

$$a b S b a \rightarrow a b a S a b a$$

表明:如果 S 是前后分别由 ab 和 ba 围着的任何串,则它才能由串 aSa 来替换。

参考文献

1. Naur P et al. Revised Report on the Algorithmic Language ALGOL 60. International Federation for Information Processing, 1962
2. Hopcroft F E, Ullman J D 著. 形式语言及其与自动机的关系. 莫绍揆, 段祥等译. 北京: 科学出版社, 1979
3. Salomaa A. Formal Languages. New York: Academic Press, 1973 (段祥)

chanshengshi biaooshi

产生式表示 (production representation)

描述一个(或一些)事件的存在导致另一事件的产生。用符号方法表述如下:

$$\text{if } A \text{ then } B$$

或

$$A \rightarrow B$$

其中 A 称为前件; B 称为后件; \rightarrow 表示由 A 为真导致 B 为真。产生式表示是最常用的一种知识表示,许多专家系统采用产生式表示来设计。

产生式源于 1943 年由 E. L. Post 提出的 Post 系统。在这个系统中定义了符号串替换的规则,其计算能力等价于 Turing 机。1965 年 Allen Newell 和 Hilbert A. Simon 将心理学的“刺激—反应”与产生式联系起来,用来建立认知模型,描述事物的因果关系。Edward A. Feigenbaum 于 1965 年—1976 年研究了 DANDRAL 专家系统,采用了产生式表示。DANDRAL 被公认为第一个专家系统。

以 Post 系统为例来说,如果 a, b 为字母,用这些字母组成有限符号行,有规则 $a \rightarrow ab$, 则有下列关系:

$$abab \Rightarrow ab^2 ab \Rightarrow ab^2 ab^2 \Rightarrow ab^3 ab^2 \Rightarrow \dots$$

如果有规则 $ab \rightarrow b$, 则

$$a^3bab^4 \Rightarrow a^2bab^4 \Rightarrow abab^4 \Rightarrow bab^4 \Rightarrow b^5$$

Chomsky 语法是一种 Post 系统, Markov 算法也是 Post 系统。

Post 规则对研究发现自然语言的语法很有用。首先,选择一种句型,将同一位置的替换词归类,得到一个词类;然后,再换一种句型,直到找到足够多的句型,将词分类完毕。然后再用乔姆斯基方法研究语法。

当描述事实时,前件可以省略,如:

→ 孔子是中国人

为了表示各种知识,对前、后件的符号串给予不同的定义,赋予不同的意义。例如:允许前件有多个项,每项是字符串、带变量的字符串、三元组(对象、属性、值)、命题、一阶谓词、数学公式等。允许用逻辑连结词 \wedge , \vee 组合各项的公式。后件一般有一项,表示结论或操作。

在实际应用中,产生式的项还可以表示图形、图像、数组、矩阵、分子结构等。产生式表示的范围非常广泛,举例如下:

```
if    (动物有翅膀)  $\wedge$ 
      (动物会飞)  $\wedge$ 
      (动物有两条细脚)
then  动物是鸟      CF = 0.85
```

其中,CF 表示可信度。

又如用三元组形式表示(在 MYCIN 系统中)

```
(细菌,染色斑,革兰氏阳性)  $\wedge$ 
(细菌,形态,球状)  $\wedge$ 
(细菌,生长结构,链形)
→(细菌,类别,链球菌)
```

在解决问题中,一个论域需要用一组产生式才能完整地刻画因果关系,在这组产生式中所使用的符号或符号串不能发生矛盾,如用符号串表示概念、事物;用产生式表示事实、规律、经验律、公理等。

一个产生式系统由三部分组成:一组产生式、一个综合数据库、一个控制系统(也称推理机)。

综合数据库包含所要解决的问题的初始数据、目标数据以及产生式执行后可能改变的用来描述中间运行状态的数据。

控制系统的运行表现为激活一串规则,并执行这些规则,每一条规则的激活条件,由综合数据库决定。执行规则会改变综合数据库,又决定下一条规则的激活条件,直到满足结束条件(或找不到可被激活的下一条规则)为止。这个过程称为推理。推理步骤大致如下:

- ① 将初始数据送到综合数据库;
- ② 由综合数据库决定激活规则的条件;
- ③ 选出所有符合条件的规则;
- ④ 根据冲突消解方法选择一条规则,并激活这条规则;
- ⑤ 执行这条规则,同时可能改变综合数据库;
- ⑥ 检查当前综合数据库是否与目标数据一致?如一致,则停机;如不一致,则执行②。

上述产生式推理是由综合数据库控制的;还有一种情况是由规则本身来控制激活条件的,这时,上述推理步骤在激活条件处要加以改动,其余基本相同。

当具有多个推理部件时,可以考虑并行推理,将不相关的推理步并行执行,提高运算效率。因推理机制是树型结构的,越靠近树根,推理的并行粒度越低,效率就越低。

HEARSAY II 是一个由美国 V. R. Lesser 等研制的语音理解系统,它发展了产生式系统,称为黑板结构。它把综合数据库推广为黑板;把一条产生式推广为一个知识源,允许由一组产生式或其他的知识表示组成;将激活产生式条件,推广为激活知识源的条件。黑板结构适合于解决复杂的专家系统控制问题。

在产生式系统中,有一种用来描述如何使用规则的产生式,称为元规则。例如,在冲突消解中将所有符合激活条件的规则排成顺序。又如,规定激活规则的条件。下面是在 MYCIN(一个医疗诊断专家系统)中使用的一条元规则:

```
if  1. (感染是骨盆脓肿)  $\wedge$ 
    2. (前件涉及肠杆菌的规则)  $\wedge$ 
    3. (前件涉及细菌是革兰氏阳性、形态是杆
        状的规则)
then 先考虑涉及 2 的规则,后考虑涉及 3 的
      规则,CF = 0.4
```

当有多个产生式均满足激活条件时,要选择一条规则激活并执行它,解决这个问题称为冲突消解。冲突消解属于控制系统的任务。针对不同情况,有多种解决方法,例如:按匹配成功的自然顺序;按综合数据库某些数据更新的记录;按运行速度;按事先安排好的优先级;按事先制定的原则等。解决冲突消解是否合理,直接影响问题的解决及计算的速度。

各国都很重视产生式系统的研究与开发,我国也做了大量有成效的工作。下面举出采用产生式表

示的有代表性的几个系统。

(1) DENDRAL 是有机化合物质谱图解析专家系统,由 Stanford 大学的 Feigenbaum 研制,创建了专家系统的设计原理和方法,产生重大的影响。

(2) MYCIN 细菌感染疾病医疗诊断专家系统,首先采用具有可信度的产生式表示,提出一套推理公式。由美国 Edward H. Shortliffe 研制。

(3) PROSPECTOR 由 Stanford 研究所的 R. O. Duda 等研制的金属矿藏勘探专家系统,在预测钨矿中,成功地获得巨大的经济效益,达到了人类专家的水平。

产生式表示的应用领域有:医学、化学、数学、计算机配置、维修、交通管理、地质勘探、测井、石油、化工、排水、环境、农业、教学、军事、模式识别、工程设计、商业、金融、管理、法律等。

产生式表示的优点有:①模块性,每个产生式是最小的知识单元,产生式之间的联结是松散的,与推理机相对独立,增加、删除产生式很方便。模块性成为建立规则库的基本论点。②易维护、易管理。③产生式形式清晰、直观,接近人的思维方式,容易被人们接受,容易实现人-机对话。

产生式表示也存在一些不足:如知识获取问题。一些领域专家很难总结出规则知识,他们的经验往往与解决具体问题混在一起,提炼成产生式本身工作量很大;还有一致性问题。目前尚难于实现自动检查产生式系统的一致性,尤其是当产生式数量很大时,可能隐含不一致,因而导致错误的结果;另外,基于综合数据库的控制不够直观,调试产生式系统有困难。

产生式表示是一种用途广泛的知识表示,尚有许多研究工作,例如:具有可信度的产生式推理及可信度的传播;具有模糊谓词的模糊产生式系统;用于控制的实时产生式系统;模糊产生式与神经网络的结合;提供实用的、开放的产生式系统开发工具;分布式产生式系统;非常大型的产生式系统;还有产生式系统的一致性;无限产生式;知识获取等。

参考文献

1. Hayes-Roth F, D A Waterman, D B Lenat. Building Expert Systems. Reading, MA: Addison-Wesley, 1983

2. 林尧瑞,张钺,石纯一. 专家系统原理与实践. 北京:清华大学出版社,1988

3. 王树林,袁志宏. 专家系统设计原理. 北京:科学出版社,1991 (王树林)

changliang

常量 (constant) 可在编译时刻确定其值,且一经确定便不能更新的计算对象。

常量可在程序内部创建并使用。可借助常量定义来定义常量,以确定其值,常量定义的作用是将一标识符绑定于一值。该标识符即为该常量的名。例如,在 Pascal 语言中,常量定义具有如下形式,即 $\text{const } I = E$,其中 I 为标识符, E 为可在编译时刻确定其值的表达式。当编译程序处理到该常量定义时,以 I 为名的常量便具有当时 E 之值,此值一经确定,在相应程序中便不能改变。常量定义的作用既有利于避免书写错误,又可节约存储空间。

参考文献

Watt D A. Programming Language Concepts and Paradigms. Prentice Hall, 1990 (徐家福)

changshi

常识 (commonsense) 人类社会中长期验证使用,众所周知的知识。

从人与自然的关系来说,常识应是那些经过人类社会千百年实践检验固定下来的,对于大多数情况而言正确地反映了对客观世界规律的认识。常识有可能有例外,但在没有迹象表明当前情况属于例外时,按常识处理,一般应是合理的、正确的。

从人与人的关系来说,常识应是一种公共性的、约定性的知识,一种由于其“众所周知”而无需在每次交往中显式声明的知识。因此,常识使人际交往更简便、经济。

为了使人与计算机的交互尽可能地像人际交往一样简便、经济,为了使计算机求解问题的知识环境更有利于问题的求解,必须考虑使计算机具有常识,包括常识内容和常识机制。这方面的研究构成人工智能理论研究的一个重要方向——常识推理。

常识推理的研究始于 20 世纪 70 年代末。对数据库和逻辑程序的语义研究揭示了所谓“封闭世界假设”(CWA)。一些学者通过分析 CWA 确立了一类常识推理手段——非单调推理。其代表人物有 McCarthy, Reiter, McDermott 等。其中 McCarthy 在倡导常识推理和常识研究方面起了关键的作用。

常识内容由于其“自明”性,一般都是隐含的,往往只有在具体建造知识系统的过程中,在人和计算机的强烈对比下才意识到它的存在。比如说,只有在计算机关于“绳子”的推理犯了一个毫无道理

的错误时,我们才意识到原来计算机需要知道“绳子只能拉,不能推”。把最起码的常识装进计算机并使之发挥作用,是一项巨大复杂的系统工程。例如美国 MCC 公司的 CYC 工程,Stanford 大学的“*How things work*”工程等。

常识机制的研究是人工智能中困难的、也是最具理论挑战性的课题之一。常识一般是有例外的,而例外情形成立与否一般需要通过推理来确认。常识机制的核心问题是在有例外、含矛盾以及知识不充分的情况下产生合理的结论。例如,“鸟会飞”,“企鹅是鸟”,“企鹅不会飞”是三件公认的“常识”。它们放到一块儿是不协调的,但在实际的推理过程中,它们都被当作常识使用,只不过推出来的结论强弱不同。遇到矛盾时,“弱”的要服从“强”的。关于常识机制的研究,典型的代表是各种非单调逻辑、次协调逻辑以及各种真值(理由)维护系统,还有面向情境、变化和动作的各种推理系统。

目前,常识主要是用逻辑公式表示。当然,最方便的形式是人类的自然语言,但要使用自然语言表示的常识可被计算机使用,就要使计算机具有一定的自然语言理解能力。CYC 工程中就采用了自然语言理解与常识内容获取两方面的研究并进的做法。

常识推理不仅涉及规则的层次,也涉及控制的层次。常识机制的最大好处在于它能按照具体情况大致上正确地快速限定搜索空间并在限定范围内进行问题求解。目前,在控制策略的层次上研究常识机制的工作还不是很多,有待进一步深入。

常识内容和常识机制是一个统一体。如果没有一套有效的常识机制,常识内容越多就越难处理;如果没有面向世界的足够多的起码常识,常识机制再漂亮也只能是玩具。就目前的研究现状看,二者要走向统一还要经过相当艰苦的努力。

参考文献

1. McCarthy J. Circumscription—A Form of Non-monotonic Reasoning. *Artificial Intelligence*, 1980, 13: 27 ~ 39
2. McCarthy J. Applications of Circumscription to Formalizing Commonsense Knowledge. *Artificial Intelligence*, 1986, 28: 89 ~ 116
3. Reiter R. On Closed World Data Bases, In: Gallaire and Minker (Eds.). *Logic and Data Bases*. New York: Plenum, 1978. 119 ~ 140 (白硕)

changweifen fangcheng shuzhi jiefa

常微分方程数值解法 (numerical solution of ordinary differential equation)

根据给定的定解条件,研究如何采用有效的数值方法将微分方程离散化以及求离散化方程的近似解的方法和过程。按定解条件的不同,可分为常微分方程初值问题和边值问题两类。

一阶常微分方程的初值问题是求函数 $y(x)$, 使满足条件

$$\left. \begin{aligned} y'(x) &= f(x, y(x)) & (a < x < b) \\ y(a) &= y_a \end{aligned} \right\} \quad (1)$$

数值解法的基本思想是:先取自变量 x 的一系列离散点

$$a = x_0 < x_1 < \cdots < x_{N-1} < x_N = b$$

对区间 $[a, b]$ 作剖分,将微分方程离散化,求出离散问题的数值解,并将其作为微分方程问题的解 $y(x)$ 的近似。例如,取步长 $h > 0$, 令 $x_n = a + nh$, 把微分方程离散化成差分方程。利用初始条件解此差分方程,得到解 $y_n, n = 0, 1, \cdots, N$ 。将 y_n 作为 $y(x_n)$ 的近似值,量 $\varepsilon_n = y(x_n) - y_n$ 就是近似解的误差,称为全局误差。数值解法的基本内容包括设计各种离散化方法,求出相应差分方程的解,估计解的误差并研究数值方法的收敛性和数值稳定性等问题。

常用的离散化方法有三种:

(1) 基于数值微分的方法 将方程 (1) 右端的导数 $y'(x)$ 用某个数值微分公式代替,得到相应的差分方程。例如,在 x_n 点用 $(y_{n+1} - y_n) / h$ 代替 y'_n , 得到欧拉向前公式

$$\left. \begin{aligned} y_{n+1} &= y_n + h f(x_n, y_n) \\ y_0 &= y_a \end{aligned} \right\} \quad (n = 0, 1, \cdots, N-1) \quad (2)$$

同样,在 x_{n+1} 点以 $(y_{n+1} - y_n) / h$ 代替 y'_{n+1} , 得到欧拉向后公式。

$$y_{n+1} = y_n + h f(x_{n+1}, y_{n+1}) \quad (2)'$$

(2) 基于数值积分的方法 在区间 $[x_n - ih, x_n + jh]$ 上对问题 (1) 中的微分方程进行积分,得到公式

$$y(x_n + jh) = y(x_n - ih) + \int_{x_n - ih}^{x_n + jh} f(x, y(x)) dx \quad (3)$$

式中的被积函数是 x 的函数。若利用某些结点上的 f 值构造此被积函数的近似函数,并且求出此积分,

便得到各种差分方程。特别地,若取 $i = 0, j = 1$, 并用 f 在结点 $x_n, x_{n-1}, x_{n-2}, \dots$ 上的值构造的插值函数代替式(3)中的被积函数,就得到亚当斯外推公式。四阶亚当斯外推公式为

$$y_{n+1} = y_n + \frac{h}{24}(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}) \quad (4)$$

式中, $f_n = f(x_n, y_n)$ 。

(3) 基于泰勒展开的方法 在设计一个计算 y_{n+1} 的算法时,在算法的公式中安排一些待定的常数。在函数光滑的假定下,将公式进行泰勒展开,并与解 $y(x_n + h)$ 的相应展开式中 h 的同幂次项相比较,按照要求的精度阶得到待定常数应满足的一些方程。求解这些方程确定待定常数,即可得到算法的公式。由此法可导出龙格-库塔公式。

龙格-库塔公式的一般形式为

$$y_{n+1} = y_n + \sum_{i=1}^k b_i K_i \quad (5)$$

式中

$$K_i = h f(x_n + c_i h, y_n + \sum_{j=1}^k a_{ij} K_j), \quad i = 1, 2, \dots, k$$

a_{ij}, c_i, b_i 为待定常数。最常用的显式四阶龙格-库塔公式为

$$y_{n+1} = y_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \quad (6)$$

式中

$$K_1 = h f(x_n, y_n)$$

$$K_2 = h f\left(x_n + \frac{1}{2}h, y_n + \frac{1}{2}K_1\right)$$

$$K_3 = h f\left(x_n + \frac{1}{2}h, y_n + \frac{1}{2}K_2\right)$$

$$K_4 = h f(x_n + h, y_n + K_3)$$

按照计算 y_{n+1} 时所用结点上值的多少可以将算法分为单步法和多步法。单步法是指已知结点 x_n 上的值 y_n 便可计算出 y_{n+1} 值的方法。多步法是指已知结点 $x_n, x_{n-1}, \dots, x_{n-k+1}$ 的值 $y_n, y_{n-1}, \dots, y_{n-k+1}$ ($k \geq 2$), 才能计算出 y_{n+1} 值的方法,这也称为 k 步方法。对于多步法,给出 y_0 后,需要由其他算法计算出 y_1, y_2, \dots, y_{k-1} 后(起步),才能使用多步法逐步计算 y_k, y_{k+1}, \dots 的值。

为了使构造的数值解法具有实用价值,需要研究算法的相容性、收敛性、误差估计和数值稳定性等问题。

相容性 将微分方程离散化成差分方程,并将微分方程的解代入该方程,出现的误差称为局部截断误差。当 $h \rightarrow 0$ 时,局部截断误差趋于零,则称该

差分方程与微分方程具有相容性。这表示差分方程是微分方程的一种近似。若局部截断误差对 h 的展开式的首项为 ch^{p+1} , 其中 c 为常数,则称局部截断误差的阶为 $p+1$, 而称相应的算法是 p 阶的。 p 越大,表示离散化后的方程与原微分方程的近似精度愈高。

收敛性 指当 $h \rightarrow 0$ 时,全局误差 $\varepsilon_n \rightarrow 0$, 即离散问题的解 y_n 收敛于微分方程的解 $y(x)$ 。这是由数值方法得到的近似解可用的理论基础。对于 p 阶算法,当 $h \rightarrow 0$ 时,误差 ε_n 将以 h^p 的速度收敛到零。 p 为误差主项中步长 h 的幂次。

误差估计 应用数值方法时最关心的问题是全局误差 ε_n 的估计。理论估计通常只能给出 $h \rightarrow 0$ 时的误差阶。一般采用事后估计法,即根据计算过程中获得的关于解的信息进行误差估计。例如通过对不同步长的计算,采用理查森外推法估计误差。利用得到的误差估计便可对数值解进行校正。

数值稳定性 指在计算过程中,某一步上产生的误差一步一步传递下去不会影响到数值解的精度。在研究数值稳定性时,微分方程

$$y' = \lambda y \quad (7)$$

常常作为研究微分方程的代表,其中 $\lambda = \alpha + i\beta$, $\alpha < 0, i = \sqrt{-1}$ 。许多数值稳定性的定义都以这个方程为基础给出的。在研究常微分方程数值稳定性时,首先需要研究稳定区域,这可参见文献[1]。

对于常微分方程组初值问题

$$y' = f(x, y), \quad y(0) = g$$

式中, $y = (y_1, y_2, \dots, y_m)^T, f = (f_1, f_2, \dots, f_m)^T, g = (g_1, g_2, \dots, g_m)^T$ 都是 m 维向量,若微分方程组右端函数 $f(x, y)$ 的雅可比矩阵 $\frac{\partial f(x, y)}{\partial y}$ 的特征值 $\lambda_j = \alpha_j + i\beta_j, j = 1, 2, \dots, m$, 满足 $\alpha_j < 0$, 并且比值

$$S = \frac{\max |\operatorname{Re} \lambda_j|}{\min |\operatorname{Re} \lambda_j|} \gg 1$$

称这类问题为刚性问题。对刚性问题应用通常的显式方法^[1]时,由于数值稳定性, $|h\lambda_j|$ ($j = 1, 2, \dots, m$) 受到限制, h 必须取得很小,因而计算量非常大。向后公式、梯形公式都是数值求解刚性问题的有效方法。

各种实际问题导出的不同类型常微分方程边值问题中,较简单的是两点边值问题:求函数 $y(x) = (y_1(x), y_2(x), \dots, y_m(x))^T, x \in [a, b]$, 使其满足微分方程组

$$y'(x) = f(x, y) \quad (8)$$

和边值条件

$$g(a, y(a), b, y(b)) = 0 \quad (9)$$

式中 m 维向量函数 f 和 g 都是已知函数。条件(9)是在两个端点上对解的约束条件。有些实际问题要求解在区间 $[a, b]$ 的多个点上的值满足预定的条件,称这样的问题为多点边值问题。

解常微分方程边值问题的数值解法有差分法和打靶法。

差分法 直接将微分方程离散化,加上边值条件构成一个代数方程组,解此代数方程组即可得到边值问题的数值解。

打靶法 其基本思想是将边值问题化成一系列初值问题求解,适当选择和调整初值 $y(a)$, 使得相应的解 $y(x)$ 满足指定的边值条件。下面以两点边值问题为例进行说明。若已知初值 $y(a)$ 的值,求解初值问题可确定解 $y(x)$, 从而惟一确定 $y(b)$ 。因此, $y(b)$ 是初值 $y(a)$ 的函数,记为 $y(b) = p(b, y(a))$ 。将 $p(b, y(a))$ 代入式(9)中的 $y(b)$, 得到只含未知量 $y(a)$ 的代数方程组

$$g(a, y(a), b, p(b, y(a))) = 0 \quad (10)$$

若将 $y(x)$ 看成是弹道,将 $y(a)$ 看成是射击条件,则 $y(b)$ 就是在射击条件 $y(a)$ 下所达到的靶子。因此打靶法的求解过程是不断地调整射击条件 $y(a)$, 使它和相应的靶子 $y(b)$ 满足预定的边界条件。若 b 固定,从数学上看,打靶法是要求解 $y(a)$ 的方程(10)。因此求解非线性方程的各种数值方法都可以应用到常微分方程边值问题。例如,应用牛顿方法,得到牛顿打靶法,其计算步骤如下:

步骤1 适当选取 $y(a)$ 的迭代初值 $y^{(0)}(a)$, 令 $i = 0$ 。

步骤2 数值求解常微分方程初值问题

$$y'(x) = f(x, y(x))$$

$$y(a) = y^{(i)}(a)$$

将得到的解记为 $y^{(i)}(x)$, 特别得到 $y^{(i)}(b) = p(b, y^{(i)}(a))$ 。

步骤3 计算

$$g^{(i)} = g(a, y^{(i)}(a), b, y^{(i)}(b))$$

若满足精度要求,即有 $\|g^{(i)}\| < \delta$, 则停止计算,取 $y^{(i)}(x)$ 为所要求的近似解;否则转步骤4。 δ 为预先给定的精度常数。

步骤4 计算

$$y^{(i+1)}(a) = y^{(i)}(a) - G_i^{-1} g^{(i)}$$

式中

$$G_i = \left[\frac{\partial g}{\partial y(a)} + \frac{\partial g}{\partial y(b)} \frac{\partial p}{\partial y(a)} \right] \bigg|_{y(a)=y^{(i)}(a)}$$

步骤5 令 $i \leftarrow i + 1$, 转步骤2。

在实际实现时,可将上述算法作各种变形和简化。

目前,数值求解常微分方程已有许多著名的软件包,如 LSODE 和 RADAU,前者采用亚当斯内插法和向后微分方法,可求解刚性和非刚性问题,运行过程中能自动变阶和变步长,后者采用隐式龙格-库塔方法。

参考文献

1. Gear C W 著. 常微分方程初值问题的数值解法. 费景高, 刘德贵, 高永春译. 北京: 科学出版社, 1978
2. Hairer E, Norsett S P, Wanner G. Solving Ordinary Differential Equations I, Nonstiff Problems. Berlin: Springer-Verlag, 1987
3. Hairer E, Wanner G. Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems. Berlin: Springer-Verlag, 1991 (费景高)

chaodao jicheng dianlu

超导集成电路 (superconducting integrated circuit)

将一定数量的超导元件和约瑟夫逊结器件做在一块芯片上以完成一定的电路功能的集成电路。当一些金属、合金和化合物冷却至低于它们的临界温度 T_c 时, 它们的电阻陡然降为零。如果是一个超导回路, 则其中的电流将会无限制地流动, 除非升高温度或外加磁场, 才能中断该超导电性, 具有这样性质的物体即称之为超导体。迄今为止, 超导体可按其 T_c 粗略地划分为两类: 一是经典低 T_c 金属性超导体, 主要为 Nb (9.5K), NbN (15K), Nb₃Sn (21K), Nb₃Ge (23.2K); 另一是高 T_c 氧化物超导体, 目前公认的为 Y₁Ba₂Cu₃O_{7-x} (91K), (Bi, Pb)₂Sr₂Ca₂Cu₃O_{10+y} (110K), Tl₂Ba₂Ca₂Cu₃O_{10+δ} (125K)。

若由两个超导体构成一个隧道结, 它们之间用绝缘层分开, 当此绝缘层减薄到一定程度, 且将此冷却到临界温度以下时, 将有电流流过此绝缘层, 此即为隧道效应超导电流。它与一般隧道效应不同之处在于不需施加电压, 这种超导隧道效应称为约瑟夫逊效应, 据此制成约瑟夫逊结器件。当电流超过此超导电流的阈值时, 器件以极快的速度由零电阻态转变为高电阻态。由于该超导阈值电流是穿透约瑟夫逊结有效面积的函数, 故可用施加外磁场来减

小阈值电流,从而使约瑟夫逊结器件由超导态(零电压)转变为正常态(有限电压)。这个电压变化值约在 3 mV 左右(而硅(Si)和砷化镓(GaAs)晶体管的转换电压在 77 K 时约为 200 mV 和 500 mV),其开关速度可达 0.1 ps,工作电流一般低于 1 mA,功耗低于 1 μ W。功耗与开关时间的乘积通常作为比较数字技术的品质因素,约瑟夫逊结的此乘积比目前任何半导体工艺得到的数值小 2~3 个数量级。但约瑟夫逊结器件没有隔离,没有确定的增益和非逆置性,从而增加了超导集成电路设计的复杂性。超导集成电路的制备与半导体集成电路有许多相似之处,其中制版、光刻、蒸发和溅射工艺基本相同,它无需离子注入、高温扩散和外延淀积工艺,而且约瑟夫逊结器件不用衬底,片子可做得比用半导体工艺的大。目前超导逻辑电路和数字电路较为成熟,已有超导模拟数字转换器、超导移位寄存器和超导动态随机存取存储器。它所用的工作致冷剂为液氮(4.2 K)。上述经典低 T_c 金属性超导体,主要是用铌(Nb)制成的。

利用高 T_c 氧化物超导体制作集成电路的工作已大力开展。目前已研制成高温超导晶体管,它的输入电阻小,输出电阻大,可作为连接固态电子元件和低温超导器件的桥梁,克服了约瑟夫逊结器件由于输出电阻小、输出电压低,很难与半导体器件匹配连接的困难。高温超导晶体管是一种反向场效应晶体管,它的输出电压是由输入电流控制的,在液氮温度(77 K)下工作。可用它制成放大器、振荡器、相移器和混频器等多种电路。

近来提出的一种快单磁通量子(RSFQ)逻辑电路不需要隧道结,而只需用一般的弱连接或高 T_c 超导体与正常金属结(SNS)来组成。它在 77 K 时的性能优于半导体电路,简单的 RSFQ 电路其时钟速度已超过 100 GHz,期望可达 300 GHz 以上。另一优点是功耗小,4 K 时为 100~500 nW/门。在 40~50 K 时,预期功耗将增加一个数量级,可用于制成各种超高速数字器件,如模数转换器、移位寄存器、数字处理器等。现已利用以 SrTiO_3 为衬底的 $\text{Y}_1\text{Ba}_2\text{Cu}_3\text{O}_{7-x}/\text{Ag}/\text{Pb}$ 合金组成的多层薄膜制成邻近效应器件,把三个特性几乎相同的器件用超导导线相连构成一个简单的逻辑电路,它具备逻辑加和逻辑乘的功能。RSFQ 逻辑电路的近期应用是制造低热耗的模数转换器。

将低 T_c 超导集成电路芯片与半导体集成电路芯片通过高 T_c 超导传输线有效地连接起来组成具

有特定功能的超导-半导体混合集成电路,可在不同温度下工作,它将是超导集成电路实用化的发展方向。

参考文献

Hara K. Superconductivity Electronics. Prentice-Hall, Inc., 1987 (徐鸿达)

chaoji xiaoxing jisuanji

超级小型计算机 (super-minicomputer) 字长不小于 32 位的一类高性能小型计算机。其处理能力、内存储器和外存储容量、操作系统功能等方面都远超过一般小型计算机。

超级小型计算机是于 1977 年在小型计算机的基础上发展而成的。它的字长为 32 位,因此,可以有 32 位的寻址空间,并增加了计算精度。一般超级小型计算机与它原来的小型计算机软件兼容。复杂的指令系统中包含了所兼容的小型计算机的指令集。超级小型计算机可配置大容量的主存储器和容量的外存储器,采用多条高速 I/O 总线以获得高的 I/O 传输速率,可支持并行处理和分布式处理,且具有较高的性能价格比。

超级小型计算机由中央处理器、高速系统总线、存储器和输入输出子系统组成。中央处理器由大规模集成电路或超大规模集成电路构成,有中断和异常处理机制,有 32 位主数据通路、高速缓冲存储器、浮点处理器、地址变换机构、指令缓冲器等。还有 ROM、EPROM 用于系统初始化、诊断程序、系统加载和引导。存储器由存储控制器和存储模块组成。高速系统总线作为存储器和中央处理器之间的高速通道,允许 64 位的读写操作。I/O 子系统由 I/O 总线、系统总线接口和外围设备组成。

超级小型计算机的早期典型代表是 VAX-11/780,其虚拟地址扩充到 32 位,采用 16 个 32 位通用寄存器;数据类型增加了 64 位双精度浮点数以及扩展精度等,同时增加了可变位字段数据格式和字符串以及十进制数格式。VAX-11/780 的出现,改变了 16 位小型机的结构,其性能达到了当时中型计算机的水平。

超级小型计算机的应用领域十分广泛,它可作为集中式的部门级管理计算机,在大型应用系统中作为前端处理机,在客户-服务器结构中作为服务器(例如作为文件服务器、数据库服务器、通信服务器和应用服务器等),具有多处理器结构的超级小型计算机可用于并行处理。

(方金仰)

chaojiedian jiegou

超结点结构 (supernode architecture) 大规模并行处理系统中支持物理上单一地址空间并由硬件实现高速缓冲存储器一致性的共享存储多处理机结构。

为了提高可扩展性,大规模并行处理系统多采用分层结构,整个系统被分成不同层次,由不同类型的网络互连,支持不同的存储结构。目前主要有**共享存储和消息传递**两类不同的结构层次,超结点对应共享存储结构层次。在一个由多个超结点互连构成的并行系统中,每个超结点拥有单独的共享物理地址空间,超结点间通过消息传递交换数据;超结点内各处理机间拥有单一的物理地址空间并由硬件实现各处理机高速缓冲存储器的一致性,支持各处理机间的数据共享。

超结点结构的主要特点是:①具有由多个处理机,由硬件实现物理上的单一地址空间和高速缓冲存储器一致性;②只有一个操作系统副本;③软件主要采用共享存储编程模式;④任一处理机都能够访问超结点中所有的存储位置和输入输出设备。

超结点结构主要有3类,即对称多处理机(SMP)结构、高速缓冲存储器一致性非均匀存储器访问(CC-NUMA)结构和唯高速缓冲存储器存储结构(COMA)。

(1) SMP结构 在该结构系统中,多个处理机通过高速侦听总线或定制的交叉开关网络与构成单一地址空间的所有存储模块互联,各处理机可直接访问各存储模块的任意位置,且所用存储访问时间相同。因此,SMP属于集中存储的均匀存储访问(UMA)结构。SMP中的高速缓冲存储器一致性由系统总线硬件通过总线侦听协议实现,对称性体现为各处理机对系统中存储器、外部设备和操作系统服务的访问是对等的。

(2) CC-NUMA结构 为提高可扩展性,CC-NUMA结构中存储器是物理分布的,各处理机都有本地存储器并通过系统总线直接相连,一个处理机的本地存储器成为其他处理机的远程存储器并通过定制的实现高速缓冲存储器一致性的**互联网络**访问,各处理机对本地存储器和远程存储器的访问时间不同,访问远程存储器的延迟比访问本地存储器大得多,达一到两个数量级。高速缓冲存储器一致性通过硬件实现的基于高速缓存目录的一致性协议来保持。由于不同存储器访问时间不同,系统软件必须尽可能保证数据

的局部性,如采用页面迁移。

(3) COMA结构 COMA结构是在CC-NUMA结构上的一种演变,它将CC-NUMA结构中的所有局部存储器都组织成高速缓存,称为COMA高速缓存,因此COMA的主存储器是由各COMA高速缓存构成的。另一个主要变化是在COMA结构中为保证数据局部性而做的数据迁移是由硬件实现的,而不是操作系统。数据迁移的单位不再是页,而是高速缓存的行。

目前广泛采用的超结点结构是SMP和CC-NUMA,最大系统处理机个数分别为64和512,COMA由于过于复杂而少有真实系统。通过引入分区概念增加操作系统副本使得超结点结构的可用性问题的解决。未来伴随多处理机芯片的问世,芯片本身就可较好地支持直接互连成SMP或CC-NUMA结构。

参考文献

1. Kai Hwang, Zhiwei Xu. Scalable Parallel Computing: Technology, Architecture, Programming. McGraw-Hill Companies, Inc., 1998
2. 卢锡城. 关于大规模并行处理机系统可扩展性设计. 中国工程科学, 2000 (孟丹)

chaowenben

超文本(hypertext) 一种非线性网状链接结构的文本。由于现代大多数多媒体文档都基于超链接结构,这种结构的多媒体文档也常称为**超媒体**,超媒体是超文本的拓展形式。

超文本与以线性形式进行组织的文本有很大不同。它以结点(也称为部件)为单位组织各种信息,一个结点是一个“信息块”,结点内的信息可以是文本、图像、图形、动画、声音或其组合;结点间通过链加以链接,形成一个网状结构。链可以指向文档中的任一部分,也可以指向另一文档。链附属于锚,锚可以是一个字或一个句子。链包含访问目标文档的所有必需的信息,链是可激活的。

超文本中的非线性文档结构思想最早由V. Bush在1945年提出。“超文本”这个术语由T. Nelson于1965年发明。当时他开发了一个真正的超文本系统——Xanadu。D. Englebart是超文本研究的另一先驱,他于1968年开发成功交互式多用户超文本系统NLS。

超文本的最大优点是提供了非线性的信息连接。与普通读物不同,超文本不仅可以顺序地阅读,

而且可以选择阅读路径,它提供了自然有效的信息导航。超文本的研究重点是关系管理。超文本中另一个重要的概念是“访问部件”,它指获得部件数据的直观效果,例如文字、图形的视觉效果,声音的听觉效果等。超文本中对于部件访问的重要特征是,访问的时间完全由用户自己决定,触发另一个链接或结束应用程序则终止部件访问。

多媒体是随着计算机对音频和视频的支持技术发展起来的。多媒体主要以演示的形式表现出来,这与基本的超文本有很大的不同。多媒体演示中所包含的部件是按某种规定的次序出现的,因而在多媒体文档中必须引入一个重要的概念——时间。当然,用户仍有控制部件访问或不被访问的能力。但是,正因为有了时间的概念,从而即使在没有用户干预的情况下,演示仍然会发生变化。所以在**多媒体文档**中,不仅要定义出部件之间的位置关系,而且也需要定义出它们之间的时间关系,即多媒体对象必须包括时空关系。

超媒体是在超文本和多媒体发展到一定的阶段的产物。超媒体包括了超文本和多媒体两者的所有特点,也就是说,一方面,它要支持结点、链接等复杂的浏览关系,同时,也要引入多媒体表现,尤其是时间的概念,因为超媒体中的信息最终是以多媒体演示的形式呈现给用户的。作为一个较为严格的超媒体的定义,强调文档中不仅要有多种类型的媒体数据,还要具有时间合成的概念,以体现多媒体的特性。

(徐光祐 潘志庚 史元春)

chaowenben zhibiao yuyan

超文本置标语言 (hyper text markup language, HTML) 一种描述超文本的置标语言。

超文本中既包含文字信息又包含“超链接”信息。一般而言,超链接是用于表达文档内不同信息元素之间相关性的一种手段。当两个信息元素在信息内容上有相关性,而它们又位于文档的不同位置,使用超链接便可以把它们联系起来,虽然文档物理位置的距离较远,也能上下文参照。在 Internet 上,超链接则运用统一资源定位地址 (URI) 定位标记,把两个网上位置不同的信息元素联系起来,在浏览网页信息时,可以连续地浏览分布于全球的网上信息。URI 是万维网联盟 (W3C) 制定的一种用于信息资源标识的标准格式。针对因特网上全球的信息资源,它对每一个共享的信息资源赋予一个惟一标识的 URI 名。超链接使用 URI 指定所要参照的信息

资源,不管它涉及的网页储存在世界上何处,通过因特网都能够链接到那个网页信息元素。

W3C 国际组织关于 HTML 语言的第一个推荐标准 HTML 3.2 是于 1997 年 1 月发布的。HTML 语言包括了描述网页和各种信息资源的语言成分: ①描述正文、表、标题、表格和图片等信息元素; ②在网页中放置超链接; ③数据表单,通过和附本程序的连接,可以向数据库服务器等服务设施提出查询或处理要求,并获得结果数据; ④在网页文档中嵌入视频和音频等多媒体信息等。

通过 HTML 语言和程序附本可以创建动态万维网页面或应用信息系统。适合 HTML 的附本编程的主要技术是文档对象模型 (DOM),它是一种独立于平台和编程语言的应用编程接口。

样式表用于控制网页文档的形象化呈现。用于 HTML 文档的样式表描述语言有 CSS。

随着可扩展置标语言 (XML) 的发展,传统的 HTML 语言也在变化,向新一代 HTML 平滑过渡。

参考文献

1. W3C. HTML 4. 01 Specification. December 1999. <http://www.w3.org/TR/1999/REC-html401-19991224/>
2. W3C. Cascading Style Sheets, level 2 (CSS2). W3C Recommendation. <http://www.w3.org/TR/1998/REC-CSS2-19980512/>
3. W3C. Cascading Style Sheets, level 1. December 1996 <http://www.w3.org/TR/REC-CSS1-961217.html> (瞿裕忠)

chenshuxing biaooshi

陈述性表示 (declarative representation)

一种描述事物的属性及事物间相互关系的知识表示方法。多数以逻辑表示的形式出现,它强调知识的静态方面。

陈述性知识本身并没有给出怎样运用知识的方法,因此任何陈述性知识如果要被实际使用,必须有一个相应的过程去解释、执行它,这种过程一般是隐含在系统之中而不是面向用户的。

下面给出两个用陈述性表示方法表示知识的例子。

一个是简单的字母排序知识。在陈述性表示方法中通过顺序关系“A 在 B 之前,B 在 C 之前,C 在 D 之前,……,Y 在 Z 之前”可以将顺序关系显式地表示出来。这种表示很方便,当要判定两个字母的

顺序时,通过一个排序程序对次序关系进行操作,就可以得出结论。

另一个是简单的逻辑知识的例子。假定有如下事实:

$\forall x(\text{Person}(x) \rightarrow \text{mortal}(x))$

$\forall x(\text{dog}(x) \rightarrow \text{mortal}(x))$

$\text{Person}(\text{小张})$

$\text{Person}(\text{小李})$

这些事实表示,所有的人都必然会死,所有的狗也必然会死,小张和小李是人。如果给出的推理规则为肯定前件的假言推理,则人们可以很容易推出小张最终必然会死的。

陈述性表示的**知识库**具有如下特征:其中的知识是一个个独立的知识片,它们在推理机的控制下通过全局数据基(间接)相互作用来达到问题的求解,它的修改是通过各个独立的知识片的更新、增加或删除来实现的。

陈述性表示的优点:①具有模块性,每一个知识片可以相对独立地增加、删除和修改;②表示简要、自然、清晰。许多重要性质可以直接得到,而不需要推导,而且每个知识片都是知识集的一个片段,易于为用户或是系统的其他部分所理解。

陈述性表示的缺点:①执行效率低;②过程知识和启发式知识不易表达;③知识库缺乏组织原则。

参考文献

1. 陆汝钤. 人工智能(上册). 北京: 科学出版社, 1989

2. 管纪文, 刘大有等. 知识工程原理. 长春: 吉林大学出版社, 1988 (刘大有 唐海鹰)

chengyuwang

城域网(metropolitan area network) 在 5 ~ 100 km 的地理覆盖范围内,以高的传输速率支持数据、声音和图像综合业务传输的一种通信网络。它以光纤为主要传输介质,其传输速率为 100 Mb/s 或更高。美国电气和电子工程师学会(IEEE)在 1992 年 2 月正式将城域网(MAN)标准定为 IEEE 802.6 分布式队列双总线(DQDB)结构。

城域网是城市通信的主干网。它充当不同的**局域网**之间通信的桥梁,并向外连入**广域网**。城域网提供高速综合业务服务。它一般采用简单、规则的网络拓扑结构和有效的介质访问方法,避免复杂的**路由选择**和流量控制,以达到高传输速率和低差错

率。城域网还允许灵活的网络结构和站点增减。较典型的城域网有以下几种:

(1) 光纤分布数据接口(FDDI) 光纤分布数据接口除了用于局域网外,也可用于城域网,相应的国际标准为 ISO 9314。FDDI 采用具有冗余的双环结构,主环与次环的回转方向相反,标准的传输速率为 100 Mb/s。

(2) 曼哈顿街道网(MSN) 曼哈顿街道网是在 1984 年提出的,其拓扑结构是规则的方格形网格,如同整齐的道路,站点在线路交叉点上。MSN 采用偏转式路由选择算法,其实质是利用链路的多余频带将信息包储存在信道中传播,而不是由结点储存。它还有自动流量控制的能力。

(3) 混洗交换网 它仿效并行处理中使用的混洗交换开关网络(参见**互连网络**),信息包可从发送站点经过混洗交换网到达任一目标站点。它的路由选择算法比较简单,且具备改变原有路由的能力,可进行流量控制和故障恢复。

(4) 分布式队列双总线 分布式队列双总线采用两条总线,流向相反。总线的标准传输速率为 150 Mb/s,最高可达 600 Mb/s。总线按长度为 53 B 的时隙分配,由头端站产生连续的时隙流和帧结构。分布式队列双总线结构见图 1。分布式队列双总线也可组成开环总线结构,当网络某处发生故障时,可将主站环开口处闭合,而在故障造成开口处重构主站,从而使网络能继续运转。分布式队列双总线开环总线结构见图 2。

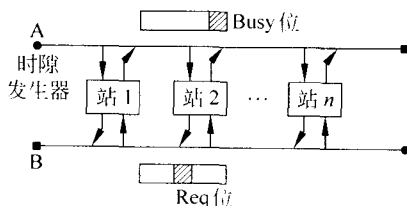


图 1 分布式队列双总线结构

分布式队列双总线的总线时隙又分为预先仲裁时隙和排队仲裁时隙,分别传送等时和非等时业务。两者由时隙中的标志区分。排队仲裁时隙由访问控制字段中的 Busy 位和 Req 位来控制对时隙的申请和占用。现以图 1 中总线 A 为例说明(总线 B 与总线 A 是对称的)。申请发送的站点从总线 B 上置 Req 位,该总线的上游站点据此统计下游站点的 Req 数。站点必须先让总线 A 上相应数目的空闲时

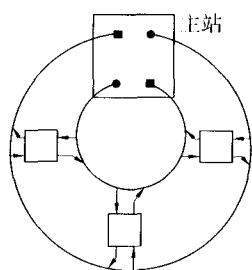


图2 分布式队列双总线开环总线结构

隙(未置 Busy 位者)流过,然后才占用此后的空闲隙发送。这样,就实现对发送请求的先进先出服务。

分布式队列双总线的帧格式与宽带综合业务数字网中采用的 ATM 帧格式(参见异步传送模式)很相似,以便分布式队列双总线向宽带综合业务数字网发展。

参考文献

1. 孙海荣等. 城域网综述. 电信科学, 1993, 9(11): 2~9
2. Stallings W. Local and Metropolitan Area Networks. Fourth Edition. New York: MacMillan Publishing Company, 1993 (史美林)

chengfaqi

乘法器(multiplier) 对以数字形式表示的两个 n 位数求积的一种运算电路。

两个原码数相乘,其乘积的数值为两数绝对值之积,而符号为两数符号的异或值。设 X 、 Y 为被乘数与乘数, X_s 、 Y_s 为被乘数与乘数的符号,则其乘积 $P = |X| \times |Y|$, 符号 $P_s = X_s \oplus Y_s$ 。

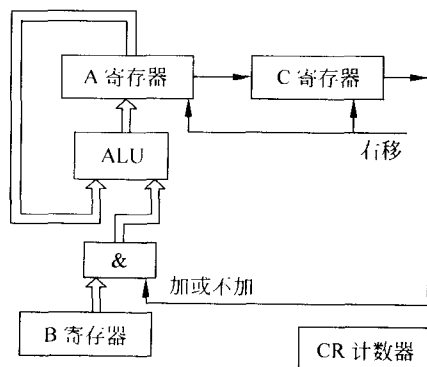
设 $X = 0.1101$, $Y = 0.1011$, 人工计算乘积 $X \cdot Y$ 的过程如下:

$$\begin{array}{r}
 0.1101 \\
 \times 0.1011 \\
 \hline
 1101 \\
 1101 \\
 0000 \\
 1101 \\
 \hline
 0.10001111
 \end{array}$$

即 $X \cdot Y = 0.10001111$, 符号为正。

计算机内实现原码一位乘法的逻辑电路如图1所示,其中三个寄存器 A、B、C 分别存放部分积、被乘数和乘数,计数器用来控制累加与移位次数。原

码一位乘法流程如图2所示,其中 CR 为计数器, C_0 为 C 寄存器的最低位。运算方法如下:



$$\begin{array}{c}
 X_s \\
 Y_s
 \end{array}
 \rightarrow \begin{array}{c} \text{=} \\ \text{=} \end{array} \rightarrow P_s$$

图1 原码一位乘法逻辑电路

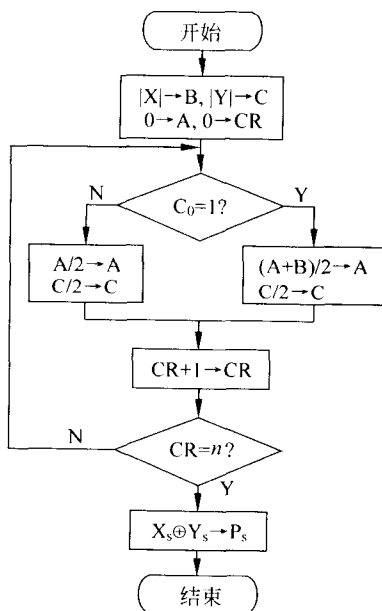


图2 原码一位乘法流程

(1) 在计算机内多个数据一般不能同时相加,一次加法操作只能求出两数之和,因此每求得一个相加数,就与上次部分积相加。

(2) 人工计算时,相加数逐次偏移 1 位,而最后的乘积位数是乘数(或被乘数)位数的 2 倍,如按这种方法在计算机中运算,加法器也要加长 1 倍。从计算机的计算过程可以发现,在求本次部分积时,前

一次部分积的最低位,不再参与运算,因此可将其右移一位,相加数可直送而不必偏移,于是用 n 位加法器就可实现两个 n 位数相乘。

(3) 部分积右移时,乘数寄存器 C 同时右移一位,这样可以用乘数寄存器的最低位来控制相加数(取被乘数或零),同时乘数寄存器 C 的最高位可接收部分积右移出来的一位,因此,完成乘法运算后, A 寄存器中保存乘积的高位部分, C 寄存器中保存乘积的低位部分。

上述的乘法运算是采用串行移位和并行相加的方法,这种方法不需要很多的硬件,然而累加-移位的方法毕竟太慢,对于 $n \times n$ 位乘法,就需要 n 次累

加和移位,即使是采用每次乘两位的乘法,将乘法速度提高了一倍,仍然嫌慢。

自从大规模集成电路问世以来,高速的单元阵列乘法器应运而生,为高速乘法的实现提供了硬件基础。另一方面,优化乘法算法,压缩累加的次数,也是提高乘法速度的好方法。图 3 示出了一个阵列乘法器,可完成 $X \cdot Y$ 乘法运算($X = X_1 X_2 X_3 X_4, Y = Y_1 Y_2 Y_3 Y_4$)。阵列的每一行送入乘数 Y 的每一数位,每一列则送入被乘数 X 的每一数位。图中每一个方框包括一个与门和一位全加器。该方案所用加法器数量较多,但内部结构规则性强,适于用大规模集成电路实现。

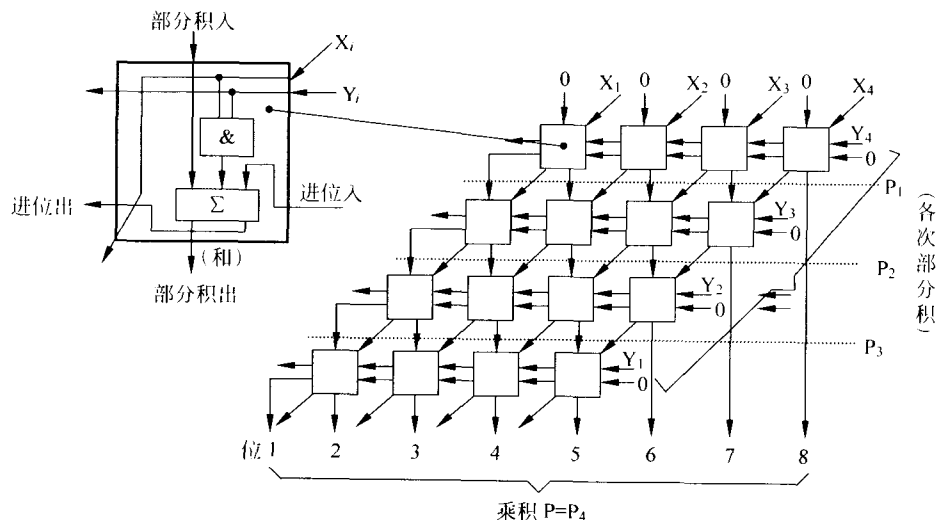


图 3 阵列乘法器

参考文献

1. 蒋本珊编著. 电子计算机组成原理. 北京: 北京理工大学出版社, 1994
2. 王爱英主编. 计算机组成与结构(第三版). 北京: 清华大学出版社, 2001 (刘恩德)

chengxu

程序(program) 计算任务的处理对象和处理规则的描述。任何以计算机为处理工具的任务都是计算任务。处理对象是**数据**(如数字、文字、图形、图像、声音等,它们只是表示,而无含义)或**信息**(赋予数据的含义)。处理规则一般指处理动作和步骤。在**低级语言**中,程序是一组指令和有关的数据或信息。在**高级语言**中,程序一般是一组**说明**和**语**

句。程序是程序设计中最基本的概念,也是软件中最基本的概念。程序是软件的**本体**,又是软件的研究对象。程序的质量决定软件的质量。以上是在实现级语言中程序的含义。在设计级语言中,程序即**设计规约**,在功能级语言中,程序即**功能规约**,在需求级语言中,程序即**需求定义**(参见**软件需求定义**)。但习惯上,程序均指实现级语言中的程序。

程序要能实际起作用,必须装入到机器内部。程序的实际工作过程称为程序的**执行**。衡量程序质量,除对程序结构进行静态考察外,还必须考察其执行过程。与执行过程无关的特性称为程序的**静态特性**;与执行过程有关的特性称为程序的**动态特性**。

发展过程

在软件发展的第一阶段(1946年—1956年),

程序都是用**机器语言**或接近于机器语言的**汇编语言**书写的,即都是低级语言程序,从内部特性上看,程序内部的工作严格依顺序执行,因此,都是顺序程序。衡量程序质量的标准主要是功效,运行时间要省,占用空间要小。在软件发展的第二阶段(1956年—1968年),程序主要都用**高级语言**书写,即高级语言程序。当然,低级语言程序仍然存在。这时除了顺序程序以外,还出现了具有并行成分的并发程序和并行程序。衡量程序质量的标准,已经逐步转向易读性和易维护性。在软件发展的第三阶段(1968年以来),由于程序的规模增大,对程序的模块化、结构化的要求越来越高,出现了一些模块化语言。同时,由于并发程序的比重增高,为了更好地书写并发程序,出现了一些具有并行成分的语言,并且由于实时处理的需要,在语言中设有相应的实时处理成分。总之,这一阶段的程序,主要是具有并行成分和实时处理成分的模块化程序,即现代高级语言程序。衡量程序质量的标准主要是结构良好性,使之易读、易维护。

基本成分

构成程序的基本成分包括**子程序**、**子例程**、**例程**、**协同例程**、**递归例程**、**模块**和**构件**,它们均称为程序单位。

子程序 与子计算任务相应的处理对象和处理规则的描述。

子例程 可由其他程序或子程序调用的子程序。子例程有两个方面:一个是定义方面,称为子例程定义或子例程说明;另一个是调用方面,称为子例程调用。随着实现方式的不同,又可区分为开式子例程和闭式子例程。二者各有利弊。开式子例程时间节省,空间浪费;闭式子例程恰恰相反。

例程 子例程的同义语。

协同例程 一组可以互相调用的程序单位,它们彼此处于平等地位,调用后毋须返回到开始位置,且自带工作区。

递归例程 可以作为其本身的子例程而被调用的例程。这种调用可以是直接的,也可以是间接的(即通过其他子例程)。

模块 具有相对独立性的一组逻辑上有关的实体。在现代**高级语言**中,有各种定义模块的方式,但其主要成分是一组**声明**和一组**语句**。

构件 具有封装性、复用性和组装性的程序单位。

参考文献

1. 徐家福. 系统程序设计语言. 北京: 科学出版社, 1983
2. Sammet J E. Programming Languages: History and Fundamentals. Englewood Cliffs: Prentice Hall, 1969
(徐家福)

chengxu fenxi

程序分析(program analysis) 对程序使用一定规则和方法推导出其程序结构和数据流程的处理过程。用以程序分析的软件叫作分析程序。

在软件**维护**过程中或软件**理解**工具中,需要分析程序,对程序的结构和数据流程进行分析。分析程序一般输出控制流图和数据流图。数据流图以图形的方法表达程序系统中信息的变换和传递过程,特别是数据被引用和被定义的情况。控制流图常用流图或抽象语法树刻画程序结构。
(郑国梁)

chengxu jishuqi

程序计数器(program counter) 程序运行中指明下一条待执行指令所在地址的一种带有计数功能的指令地址寄存器,又称**指令计数器**。当一条现行指令执行完毕的时候,程序计数器作为指令地址寄存器,其内容必须已经改变成下一条指令的地址,从而使程序得以持续运行。为此可采取以下两种办法。

第一种办法是在指令中除了规定操作数的地址外也规定了下一条指令的地址。在现行指令执行过程中将这个地址送入指令地址寄存器即可达到目的。早期以磁鼓、延迟线等串行装置作为主存储器的计算机多采用此法,因为根据本条指令的执行时间恰当地选定下一条指令的地址可以缩短读取指令的串行等待时间,从而收到提高程序运行速度的效果。

第二种办法是顺序执行指令。一个程序由若干个程序段组成,每个程序段的指令可以设计成顺序地存放在存储器之中,所以只要指令地址寄存器兼有计数功能,在执行指令的过程中进行计数,自动加一个增量,就可以形成下一条指令的地址,从而达到顺序执行指令的目的。这个办法适用于以随机存取装置作为主存储器的计算机。当程序的运行需要从一个程序段转向另一个程序段时,可以利用**转移指令**来实现。转移指令中包含了即将转去的程序段入

口指令的地址。执行转移指令时将这个地址送入程序计数器(此时只作为指令地址寄存器,不计数)作为下一条指令的地址,从而达到转移程序段的目的。子程序的调用、中断和陷阱的处理等都用类似的方法。

在随机存取存储器普及以后,第二种办法的整体运行效果大大地优于第一种办法,因而顺序执行指令已经成为当代主流计算机普遍采用的办法,程序计数器就成为中央处理器不可或缺的一个控制部件。

(张梓昌)

chengxu lilun

程序理论 (theory of programs) 研究程序的语义性质和程序的设计及开发方法的理论。主要包括程序语义理论、数据类型理论、程序逻辑理论、程序验证理论、并发程序设计理论和混合程序设计理论。程序理论和算法理论是计算机科学的两大支柱。

程序理论的基本问题是如何建立一个相对完善的理论框架,为软件的设计和开发方法提供理论依据。这个框架应能提供有效地描述程序规约的语言;应能定义可操作的变换方法以便能规约构造可执行的程序;应能给出验证程序与其规约之间一致性的机制。

程序是用程序语言编写的,研究程序的规约、变换和验证,必须首先给出程序语言的语义。这种语义用数学方法刻画程序语句的加工过程,并将其执行结果形式化。所以,程序语义也叫**形式语义**。形式语义的研究始于20世纪60年代初期,ALGOL 60是第一个明确区分语法和语义的程序设计语言。J. McCarthy, P. J. Landin, C. Strachy, D. Scott, C. A. R. Hoare 和 E. W. Dijkstra 等学者,以及爱丁堡大学和维也纳 IBM 研究中心的计算机科学家们对程序语义的研究和发展都起过重要作用。

程序语言的形式语义分为四类:①**操作语义**,模拟程序执行中计算系统的操作过程。②**指称语义**,把程序作为论域间的泛函以便刻画程序的执行数学结果。③**公理语义**,用公理化方法刻画程序与被加工数据的逻辑关系。④**代数语义**,把程序执行的结果定义为满足某种公理体系的代数结构。程序理论对形式语义的需求,促进了论域、偏序以及范畴论等数学理论的发展;而形式语义理论的研究又促进了程序语言和程序设计方法的进步,例如:在高级语言中广泛使用的过程说明和过程调用的精确概念和实现机制,就是在语义理论的研究中逐步明

确的。

程序的设计和开发理论早期研究的目标是解决程序验证问题。C. A. R. Hoare 在20世纪60年代首先提出了**程序逻辑**的理论。这个理论的基本逻辑公式形式为

$$\{\phi\} S \{\psi\}$$

其中, ϕ, ψ 是关于程序变元的逻辑表达式, ϕ 称为前置条件, ψ 称为后置条件。公式 $\{\phi\} S \{\psi\}$ 为真当且仅当:如果程序 S 执行前程序变元满足 ϕ ,则程序执行后程序变元满足 ψ 。程序设计的目标就是构造使 $\{\phi\} S \{\psi\}$ 成立的程序。C. A. R. Hoare 对程序语言的每个语句都给出了相应的逻辑规则。因此,在程序 S 给定后,可以使用这些规则证明:如果程序 S 执行前程序变元满足 ϕ ,则程序执行后程序变元满足 ψ 。

霍尔逻辑的缺点是其基本形式不能进行逻辑演算。为了克服这种缺点,人们在20世纪80年代,提出了类型论方法以求为程序的设计和开发建立更完善的理论框架(见**类型理论**)。比较典型的类型理论是由 P. Martin-Löf 建立的,称为**直觉主义类型论**(见**马丁洛夫类型论**)。它的基本思想是:精选一组类型(集合)作为程序规约,而它们的元素就是满足规约的程序;用一组规则定义类型与其元素间的隶属关系,这些规则即是从规约产生程序的变换规则,又是一阶(直觉主义)逻辑的证明规则。因此,只要对给定的规约(逻辑命题)进行证明,就可以构造出符合此规约的程序。这样,程序规约、变换、验证都寓于对规约的证明之中了。马丁洛夫理论的基本对象是 $a: A$, 其中 A 是一类型, a 是类型 A 中的元素,称为 A 的居元。这个理论定义了几种基本类型: $A \rightarrow B, A \times B, A + B, \prod_{x:A} B$ 和 $\sum_{x:A} B$, 每种类型都由一组规则定义,这些规则确定了居元和类型间的关系。这个理论的基本对象 $a: A$ 有多种解释,这些解释导致了它在程序设计和开发理论中的应用。例如: $a: A$ 可以解释为(居元: 类型)、(证明: 命题)、(程序: 规约)等。以类型 $A \rightarrow B$ 为例,它有下列两条基本规则:

$$\frac{x: A \vdash b: B}{\lambda x. b: A \rightarrow B} \quad (1)$$

$$\frac{m: A \rightarrow B \quad n: A}{mn: B} \quad (2)$$

它们有三种解释:

解释1:(居元: 类型)。规则(1)表示:若类型 A 有居元 x 可以推出类型 B 有居元 b ,则类型 $A \rightarrow B$

有居元 $\lambda x. b$ 。规则(2)表示:若类型 $A \rightarrow B$ 有居元 m 且类型 A 有居元 n ,则类型 B 有居元 mn 。其中, $\lambda x. b, mn, m$ 和 n 称为 λ -项,前两者分别称为 λ -抽象和 λ -作用(见 **λ 演算**)。类型理论的重要结论是:给定典型居元 a 和类型 A ,可以在有限步内判断 a 是否为 A 的居元,这就是类型理论的强范式化性质。

解释2:(证明:命题)。采用直觉主义逻辑的观点,一个逻辑命题为真当且仅当存在此命题的证明。若把 $a: A$ 解释为 a 是命题 A 的证明,把类型 $A \rightarrow B$ 解释为 A 蕴涵 B 。则规则(1)表示:若命题 A 有证明 x 可以推出命题 B 有证明 b ,则命题 $A \rightarrow B$ 有证明 $\lambda x. b$ 。规则(2)表示:若 $A \rightarrow B$ 有证明 m 且命题 A 有证明 n ,则命题 B 有证明 mn 。若把规则(1)和(2)中的所有“证明”(类型的居元)去掉,它们就成为关于逻辑蕴涵的自然推理规则:

$$\frac{A \vdash B}{A \rightarrow B} \quad (1')$$

$$\frac{A \rightarrow B \quad A}{B} \quad (2')$$

类似地,若把类型 $A \times B, A + B, \Pi_{x:A} B$ 和 $\Sigma_{x:A} B$ 分别解释为 $A \wedge B, A \vee B, \forall x. B$ 和 $\exists x. B$,则关于这些类型的规则就成为关于 \wedge (与), \vee (或), \forall (全称量词)和 \exists (存在量词)的逻辑推理规则。这就是 Howard 的“命题即类型”原则。

解释3:(程序:规约)。假定一阶逻辑语言作为规约语言,函数式语言作为程序语言,则 λ -项: $\lambda x. b, mn, m$ 和 n 都将表示程序,而 $a: A$ 可以解释为程序 a 满足规约 A 。规则(1)和(2)在这种情况下定义了程序与规约间的关系。规则(1)表示:若程序 x 满足规约 A 可以推出程序 b 满足规约 B ,则程序 $\lambda x. b$ 满足规约 $A \rightarrow B$ 。规则(2)表示:若程序 m 满足规约 $A \rightarrow B$,且程序 n 满足规约 A ,则程序 mn 满足规约 B 。

根据这三种解释,对程序理论的基本问题:给定规约 A ,要找出满足它的程序 a ,就可做下述新的理解:①按照解释3,规约 A 是一逻辑表达式。对它可以使用逻辑推理规则进行证明。②证明的过程就是引用逻辑推理规则的过程。③又根据解释2和3,逻辑推理规则就是程序构造规则。在证明中,引用一条规则的同时,就构造出满足被证明的某个子公式的一个程序片段。④如果规约 A 被证明为真,在证明过程中就构造出满足规约 A 的一个程序。总之,根据马丁洛夫理论,只要将作为规约的逻辑公式证明为真,就可构造出满足规约 A 的程序。因此,程

序验证作为程序理论的单独组成部分就不再需要了。这就是将类型理论作为程序设计和开发理论的典型方法。

根据这种思想,人们在20世纪80年代推出了多种不同的交互式证明编辑工具,使用这类工具时,对规约的证明是由用户完成的。用户引用开发工具中的逻辑推理规则,对规约进行证明,并构造程序。这种引用的合理性是由计算机检查的,类型理论的强范式化性质保证了这种检查的可行性。目前,这类开发工具尽管保证了所开发程序的正确性,但它们的程序设计质量与人工设计的程序质量相比差距很大,尚待提高。

20世纪70年代初期,为了保证在操作系统中多个并行执行进程的正确性,导致了并发程序理论的产生。进入80年代以来,随着超大规模集成电路技术的日臻成熟,并行和分布计算机系统得到了迅速发展,特别是互联网的出现和广泛使用,大大促进了并行程序理论的发展,使之成为程序理论的一个重要分支。

并发程序是包含多个没有因果关系的进程的进程。进程间的通信、同步和它们的并行执行是并发程序区别于顺序程序的基本操作。计算机科学中的并发概念是由 Petri 在1962年提出的。Hoare 和 Milner 在20世纪70年代后期,分别提出并发程序的 CCS 和 CSP 模型,他们把输入输出以及并行执行作为基本语法成分引入程序设计语言,并使用结构操作语义方法定义模型中基本语法成分的操作语义,开创了用代数方法研究并发程序中通信和并行行为的领域。并发程序理论研究的内容包括:如何刻画并行进程的行为,在什么情况下它们可以互相模拟,研究各种通信及同步机制,以及死锁及活性,可观察性和发散性等并发现象。并发程序理论的研究加深了人们对并发系统的认识;其主要研究成果,例如,关于通信和同步的概念,已作为基本语言成分在 Ada, Occam 和 Java 等程序语言中得到广泛应用。

进入20世纪90年代,对控制系统的研究,例如,对自动导航及核电站监测等控制系统的研究,引起了程序理论界的关注。这些系统的特点是:它们都使用计算机进行实时控制;对控制系统的安全性和可靠性要求高,控制系统的微小错误都将给经济和社会安全带来巨大的损失;系统中都存在两类不同对象:根据传统控制理论,使用微分方程刻画的连续现象和根据逻辑或代数方法,使用计算机程序

控制的离散型事件。这类系统又叫混合系统。近年来,混合系统的程序理论已成为研究热点。其基本目标是:建立混合系统的计算模型,设计描述混合系统的高级语言,探索混合系统程序的设计和开发方法。目前,在建立混合计算模型方面有三种不同的方法:第一种称为逻辑型混合计算模型方法,其基本思想是在时态逻辑基础上引入时段和切变的概念。时段用于刻画系统在时间区间上的连续变化,切变则表示系统中离散事件间的时序关系。第二种是程序设计型混合模型方法,其目标是在 CSP 及 ADA 等并发语言中引入连续变量及给定初值的微分方程,而语言中原有的通信、顺序及条件语句则用来表示系统中离散事件的关系。第三种方法是将自动机概念推广,把微分方程刻画的连续现象扩充为自动机的状态,用自动机状态转移刻画离散事件间的关系。尽管混合系统理论发展的历史不长,但它在一些重要的实时控制系统中已经得到了应用,显示出这些理论的生命力。

参考文献

1. ACM Turing Award Lectures: The First Twenty Years, 1966 to 1985. Reading: Addison-Wesley, 1987
2. Martin-Löf P. Intrusionistic Type Theory. Bibliopolis, Naples, 1984 (李未)

chengxu luoji

程序逻辑 (program logic) 描述和论证程序行为的逻辑。

由于程序和逻辑有着本质的联系,因此程序逻辑的研究一直是计算机科学最活跃的领域之一。美国 R. W. Floyd 于 20 世纪 60 年代中期把程序框图与逻辑公式相对应,提出使用逻辑描述和分析程序的想法。20 世纪 60 年代末期,英国 C. A. R. Hoare 首次给出一个程序语言的逻辑系统,提出程序部分正确性的形式验证规则(见程序验证)。20 世纪 70 年代一些科学家提出用模态逻辑和时态逻辑描述和论证程序。这些工作推动了程序逻辑的研究。

程序逻辑起源于验证程序正确性的需求,用逻辑公式描述对输入和输出信息的要求,建立逻辑公式与程序间的联系,表示为:

$$\{P\}S\{Q\}$$

其中 P 和 Q 为有关程序变元的逻辑表达式; P 称为 S 的前置条件, Q 称为 S 的后置条件。此公式表示,如果程序 S 执行前程序变量的值满足前置条件 P ,且程序终止,则程序 S 执行完成时,程序变量的值满

足后置条件 Q 。可以建立一套关于这类公式的推理规则,得到一个描述程序行为的逻辑系统,这就是著名的霍尔逻辑。这种逻辑的缺点是逻辑公式不能描述程序 S 的终止特性,因此它是讨论程序部分正确性的逻辑。如果在霍尔逻辑系统中又能证明对满足前置条件的所有输入变量,程序 S 都终止,则程序具有完全正确性。

在公式 $\{P\}S\{Q\}$ 中,逻辑表达式 P 和 Q 实际上是当作语句 S 的注释使用,对逻辑公式 $\{P\}S\{Q\}$ 不能直接进行逻辑运算,如 $\{P_1\}S\{Q_1\} \rightarrow \{P_2\}S\{Q_2\}$ 就没有意义,因此霍尔逻辑还没有把程序和逻辑统一起来。

解决的方法之一就是在逻辑系统中引入模态词来刻画程序的动态行为,用模态逻辑描述程序行为。

例如,引入模态词 $[\]$,而 $\langle S \rangle Q$ 表示当 S 终止时 Q 为真;引入模态词 $\langle \rangle$,而 $\langle S \rangle Q$ 表示存在一个时刻, S 终止且 Q 为真。这样程序 S 的部分正确性就可以表示为 $P \rightarrow [S]Q$,即 P 蕴涵当 S 终止时 Q 为真;而 $P \rightarrow \langle S \rangle Q$,则表示 S 的完全正确性。

20 世纪 70 年代中期开始,并发式程序设计逐渐成为程序理论的主要研究课题之一。而不同任务间的同步和信息交换及有关死锁等是并发程序不同于顺序程序的主要的动态特性。

在逻辑系统中,引入时态联结词描述程序的动态特性是一种自然的解决方法,例如可以引入 \Diamond , $\Diamond P$ 表示在将来某一时刻 P 真;引入联结词 \Box , $\Box P$ 表示 P 从此以后真。使用时态逻辑可以对每个程序构造出它所对应的逻辑系统,并可以在这个系统中刻画程序终止(记作 STOP)及无死锁等概念。例如程序 S 的部分正确性问题就可以表示为 $P \rightarrow \Box(\text{STOP} \rightarrow Q)$ 。它表示程序开始执行时, P 真蕴含下述论断:如果将来某一时刻程序终止则 Q 真。程序的完全正确性则可表示为 $P \rightarrow \Diamond(\text{STOP} \wedge Q)$,它表示程序开始执行时 P 真蕴含下述命题:存在某一时刻 S 终止并且 Q 真。

软件工程的一个重要方法是在软件开发前首先把程序要达到的目标即规约描述清楚,然后建立一套系统地从功能描述到执行程序的开发和检验技术。功能描述应当简洁易懂,开发方法应当有效,易于验证。程序逻辑已成为联结功能规约和开发方法及进行程序验证的一个重要手段。

参考文献

1. 陆汝钊. 计算机语言的形式语义. 北京: 科学出版社, 1992

2. Manna Z. Mathematical Theory of Computation. McGraw-Hill, 1974 (李未)

chengxu sheji

程序设计 (programming) 设计、编制和调试程序的方法与过程,或研究、开发上述方法与过程中所涉及的理论、原则、方法及技术所涉及的学科。又称编程。这里的程序一般是指实现级语言的程序。它是目标明确的智力活动。由于程序是软件的本体,软件的质量主要是通过程序的质量来体现的,程序设计工作在软件研究中的地位就显得非常重要,内容涉及有关的基本概念、工具、方法以及方法学等。

分类

按照结构性质,有**结构化程序设计**与非结构化程序设计之分。前者指的是具有结构性的程序设计方法与过程。它具有由基本结构构造复杂结构的层次性,后者反之。按照用户要求,有**过程式程序设计**与非过程式程序设计之分。前者指的是使用过程式程序设计语言的程序设计;后者则指使用非过程式程序设计语言的程序设计;按照程序的成分性质,有**顺序程序设计**、**并发程序设计**、**并行程序设计**、**分布式程序设计**之分。顺序程序设计是设计、编制和调试顺序程序的方法与过程。并发程序设计是设计、编制和调试并发程序的方法与过程;并行程序设计、分布式程序设计的含义依此类推。按照设计风格,有**逻辑式程序设计**、**函数式程序设计**、**对象式程序设计**(**面向对象程序设计**)之分。逻辑式程序设计是以逻辑子句为基本构件的程序设计;函数式程序设计是以函数为基本构件的程序设计;对象式程序设计是以对象类为基本构件的程序设计。此外,尚有**可视程序设计**、**文化程序设计**等。

基本内容

程序设计的基本概念有程序、数据、子程序、子例程、协同例程、模块和构件以及顺序性、并发性、并行性和分布性等。程序是程序设计中最为基本的概念。子程序、子例程和协同例程等都是为了便于进行程序设计而建立的程序基本单位。顺序性、并发性、并行性和分布性反映程序的内在特性。

程序设计规范是进行程序设计的具体规定。程序设计是软件开发工作的重要部分,而软件开发是工程性的工作,所以要有规范。程序设计工具包括用以书写程序的语言和为了便于进行程序设计而提供的各种专用程序等。语言影响程序设计的功效,

以及软件的可靠性、易读性和易维护性。专用程序为软件人员提供合适的环境,便于进行程序设计工作。

程序设计方法有两类。一类是全局性的,如结构化程序设计方法。它不仅要求编出的程序结构良好,而且要求程序设计过程是结构化的、层次式的、逐层降低抽象级别的。另一类则是局部性的。如子例程方法、协同例程方法等。全局性的方法与规范的关系密切,而且互有影响。

程序设计的发展可归结为从顺序程序设计到并发程序设计、并行程序设计、分布程序设计;从非结构化程序设计到结构化程序设计;从过程式程序设计到非过程式程序设计,到逻辑式程序设计、函数式程序设计、对象式程序设计,以及可视程序设计、文化程序设计等;从低级语言工具到高级语言工具。

参考文献

1. 徐家福. 系统程序设计语言. 北京: 科学出版社, 1983
2. 徐家福. 对象式程序设计语言. 南京: 南京大学出版社, 1992
3. Sammet J E. Programming Languages: History and Fundamentals. Englewood Cliffs: Prentice Hall, 1969 (徐家福)

chengxu sheji fangfaxue

程序设计方法学 (programming methodology) 以程序设计方法为研究对象的学科。它主要涉及用于指导程序设计工作的原理和原则,以及基于这些原理和原则的设计方法和技术,着重研究各种方法的共性与个性,各自的优缺点。一方面,要涉及到方法的理论基础与形成背景;另一方面,也要涉及到方法的基本架构与实用价值。程序设计方法学的另一种含义是,针对某一领域或某一领域的特定一类问题进行程序设计的指导原则和所用的一整套特定程序设计方法所构成的体系。例如,基于 Ada 程序设计语言的程序设计方法学。

作为一门学科(第一种含义),程序设计方法学目前系统性的研究成果尚不多,但由于它可对程序设计人员选用具体的程序设计方法起指导作用,而具体的程序设计方法对程序设计工作的质量以及所设计出的程序的质量影响巨大,因此,程序设计方法学研究的重要性也就不言而喻。

作为进行程序设计的指导原则和一整套特定程序设计方法所构成的体系(第二种含义),目前已出

现多种程序设计方法学。例如,各种逻辑式程序设计方法学、函数式程序设计方法学、对象式程序设计方法学等等。它们各自的利弊得失均和具体领域、具体问题以及具体环境有关。上述两种含义的关系是,第二种含义是第一种含义的基础,第一种含义是在第二种含义的基础上的总结、提高,上升到原则、原理和理论的高度。因此,这两种含义的程序设计方法学都很重要。它们既对实际程序设计工作有指导意义,又对软件的发展有较大影响。(徐家福)

chengxu sheji yuyan

程序设计语言 (programming language)

用于书写计算机程序(习惯上指实现级语言程序)的语言。语言的基础是一组记号和一组规则。根据规则由记号构成的记号串的总体就是语言。在程序设计中,这些记号串就是**程序**。程序设计语言包含三个方面,即**语法**、**语义**和**语用**。语法表示程序的结构或形式,亦即表示构成语言的各个记号之间的组合规则,但不涉及这些记号固有的以及和使用情景有关的含义。语义表示程序的固有含义,亦即表示按照各种方法所表示各个记号的特定含义,但不涉及使用者。语用表示程序与使用情景有关的含义。

语言的好坏不仅影响到程序人员使用是否方便,而且涉及到程序人员所写程序的质量。

基本成分 语言的种类千差万别。但是,一般说来,基本成分不外四种。①数据成分,用以描述程序中所涉及的数据;②运算成分,用以描述程序中所包含的运算;③控制成分,用以表达程序中的控制构造;④传输成分,用以表达程序中数据的传输。

分类 按照语言级别,有**低级语言**与**高级语言**之分。低级语言包括**字位码**、**机器语言**和**汇编语言**。它的特点是与特定的机器有关,功效高,但使用复杂、繁琐、费时、易出差错。其中字位码是计算机惟一可直接理解的语言,但由于它是一连串的字位,复杂、繁琐、冗长,几乎无人直接使用。机器语言是表示成数码形式的机器基本指令集,或者是操作码经过符号化的基本指令集。汇编语言是机器语言中地址部分符号化的结果,或进一步包括宏构造。高级语言的表示方法要比低级语言更接近于待解问题的表示方法,其特点是在一定程度上与具体机器无关,易学、易用、易维护。当高级语言程序翻译成相应的低级语言程序时,一般说来,一个高级语言程序单位要对应多条机器指令,相应的编译程序所产生的目标程序往往功效较低。

按照用户要求,有**过程式语言**和**非过程式语言**之分。过程式语言的主要特征是,用户可以指明一系列可顺序执行的运算,以表示相应的计算过程。例如,FORTRAN, COBOL, ALGOL 60 等都是过程式语言。非过程式语言的含义是相对的,凡是用户无法指明表示计算过程的一系列可顺序执行的运算的语言,都是非过程式语言。著名的例子是 PROLOG 和 RPG。

按照应用范围,有**通用语言**和**专用语言**之分。目标非单一的语言称为通用语言,例如, FORTRAN, COBOL, ALGOL 60 等都是通用语言。目标单一的语言称为专用语言,如 APT 等。

按照使用方式,有**交互式语言**和**非交互式语言**之分。具有反映人机交互作用的语言成分的语言称为交互式语言,如 BASIC 语言就是交互式语言。语言成分不反映人机交互作用的语言称为非交互式语言,如 FORTRAN, COBOL, ALGOL 60, PASCAL, C 等都是非交互式语言。

按照成分性质,有**顺序语言**、**并发语言**、**并行语言**和**分布语言**之分。只含顺序成分的语言称为顺序语言,如 FORTRAN, PASCAL, C 等都是顺序语言。含有并发成分的语言称为并发语言,如并发 PASCAL, Modula 和 Ada 等都是并发语言。含有并行成分的语言称为并行语言。考虑到分布计算要求的语言称为分布语言,如 Modula * 便是分布语言。

传统的程序设计语言大都以冯·诺依曼式的计算机为设计背景,因而又称为冯·诺依曼式语言。J. Backus 于 1977 年提出的函数式语言 FP 则以非冯·诺依曼式的计算机为设计背景,因而又称为非冯·诺依曼式语言。

主要语言举例

(1) APT——自动数控程序语言 第一个专用语言,用于数控机床加工,1956 年。

(2) FORTRAN——公式翻译程序设计语言 第一个广泛使用的高级语言,为广大科学和工程技术人员使用计算机创造了条件,1956 年。

(3) FLOW-MATIC 第一个适用于商用数据处理的语言,其语法与英语语法类似,1956 年。

(4) IPL-V——信息处理语言-V 第一个表处理语言,它可看成是一种适用于表处理的假想计算机上的汇编语言,1958 年。

(5) COMIT——麻省理工学院编译程序语言 第一个串处理和模式匹配语言,1957 年。

(6) COBOL——面向商业的通用语言 使用最广泛的商用语言,1960 年。

(7) ALGOL 60——算法语言 60 程序设计语言由技艺转向科学的重要标志,其特点是局部性、动态性、递归性和严谨性,1960 年。

(8) LISP——表处理语言 引进函数式程序设计概念和表处理设施,在人工智能领域内广泛使用,1960 年。

(9) JOVIAL——国际代数语言的朱尔斯文本 第一个具有处理科学计算、输入输出逻辑信息、数据存储和处理等综合功能的语言。多数 JOVIAL 编译程序都是用 JOVIAL 书写的,1960 年。

(10) GPSS——通用系统模拟语言 第一个使模拟成为实用工具的语言,1961 年。

(11) APL 一种提供很多高级运算符的语言,它可使程序人员写出甚为紧凑的程序,特别是涉及到矩阵计算的程序,但其实现文本到 1967 年才有定义,1962 年。

(12) JOSS——Johnniac 开放系统 第一个交互式语言,它有很多方言,曾使分时成为实用,1964 年。

(13) FORMAC——公式处理编译程序 第一个广泛用于需要形式代数处理的数学问题领域的语言,1964 年。

(14) SIMULA——模拟语言 一种主要用于模拟的语言,它是 ALGOL 60 的扩充,1966 年。SIMULA 67 是 1967 年 SIMULA 的改进,其中引进了“对象”及“类”等概念,它是第一个对象式语言。

(15) PASCAL——菲利浦自动顺序计算机语言 它是在 ALGOL 60 的基础上发展起来的重要语言,其最大特点是简明性与结构性,1971 年。

(16) PROLOG 一种处理逻辑问题的语言。它已广泛用于关系数据库、数理逻辑、抽象问题求解、自然语言理解等多种领域,1971 年。

(17) Smalltalk 一种面向对象程序设计语言。自从 1971 年出现后,曾有多种不同文本,其中使用最广的是 Smalltalk 80,其显著特点是利用“对象”,以使面向对象程序设计得到广泛应用,1971 年。

(18) C 一种使用颇为广泛的程序设计语言。它原先是为辅助开发 UNIX 操作系统而设计的,后来广泛用于研究、开发与教学,主要用于系统程序设计,同时也用于其他领域,1973 年。

(19) Ada 一种现代模块化语言,它属于 ALGOL-PASCAL 语言族,但有较大变动。其主要特征是强类型化和模块化,便于实现分别编译,提供类属设施与异常处理,适于嵌入式应用,1979 年。

除了上面列举的语言外,还有一些较为通用的语言,特别是 BASIC, PL/1, SNOBOL, ALGOL 68 等。BASIC 虽然简单、易学,使用广泛,但其中没有什么新概念,而且并不是第一个交互式语言。PL/1 的设计思想来源于 JOVIAL,其功能来源于 FORTRAN, COBOL, ALGOL 60,具有中断和表处理等设施。SNOBOL 是一种好的语言,对 COMIT 中若干概念做了明显的改进。ALGOL 68 在语言成分和描述方法方面虽有所创新,但应用尚不广泛。

发展趋势 程序设计语言是软件的重要方面。它的发展趋势是模块化、简明性、形式化、并行化和可视化。①模块化,不仅语言具有模块成分,程序由模块组成,而且语言本身的结构也是模块化的。②简明性,涉及的基本概念不多,成分简单,结构清晰,易学易用。③形式化,发展合适的形式体系,以描述语言的语法、语义、语用。④并行化,发展具有合适并行成分的并行语言。⑤可视化。

参考文献

1. 徐家福. 系统程序设计语言. 北京: 科学出版社, 1983
2. Sammet J E. Programming Languages: History and Fundamentals. Englewood Cliffs: Prentice Hall, 1969
3. Patt T W. Programming Languages: Design and Implementation. 2nd ed. Englewood Cliffs: Prentice Hall, 1984
4. Tucker A B. Programming Languages. 2nd ed. New York: McGraw-Hill, 1986 (徐家福)

chengxu yanzheng

程序验证 (verification of programs) 检验程序正确性的理论和方法。为了解一个程序是否正确地实现了预定的目标,传统程序调试方法是通过对程序输入一些规定初始数据,试验性地执行这个程序,测试其是否能产生所要的答案。如果发现有误,就检查和修改所编的程序,直至对所有规定的初始数据,都能产生预期的结果。然而,对于一般程序而言,它对不同的初始输入数据的加工过程是不同的,而且初始数据的取值范围往往又十分广泛。因此,使用调试方法穷尽程序的各种可能加工过程以确保程序的正确性,几乎是不可能实现的。所以说,调试方法只能发现程序中的错误,多一次测试正确只能说明程序可靠一点,不能说明程序正确无误。程序验证则是研究如何使用数学方法,严格证明一

个程序符合其预定目标,因而是正确无误的。

美籍匈牙利科学家 J. von Neumann 在 1947 年发表的论文中就提到程序正确性证明。美国科学家 R. W. Floyd 于 1967 年系统地提出验证程序正确性的归纳断言方法,引起了计算机科学界研究程序验证的热潮。英国科学家 C. A. R. Hoare 于 1969 年将归纳断言方法形式化,提出程序验证的公理系统。1969 年以来又陆续出现很多使用归纳断言方法或者结构归纳法的自动程序验证系统,其中以 20 世纪 70 年代中期实现的 Boyer-Moore 程序验证系统最为著名。1970 年以来还出现能辅助用户正确编制程序的实用的半自动化程序验证系统。在使用这种系统时,用户必须协助系统完成程序验证中创造性最强部分的工作。

为验证程序,必须首先将程序所要实现的目标形式化,即使用数学公式表达程序加工的初始数据的范围(称作输入谓词)和程序加工的结果(称作输出谓词)。程序断言正是对程序性质的描述,形如 $\{\varphi\}S\{\psi\}$,其中 φ, ψ 为两个谓词, S 是一个程序, φ 称为 S 的前置断言,又称输入谓词, ψ 称为后置断言,又称输出谓词。断言 $\{\varphi\}S\{\psi\}$ 称为 S 关于 (φ, ψ) 的正确性断言。它的含义是:若 S 开始执行时 φ 为真,则 S 的执行必终止且终止时 ψ 为真。程序设计的任务就是:对给定的程序规约要求 (φ, ψ) ,构造程序 S ,使得断言 $\{\varphi\}S\{\psi\}$ 为真。反之,程序验证则是对已有的程序 S ,验证它是否满足规约 (φ, ψ) 的要求。下面用一个非负整数的除法程序来说明。图 1 中, x_1 是被除数, x_2 是除数, z_1 中存放程序加工后得到的商; z_2 中存放得到的余数; y_1, y_2 是程序加工时使用的工作单元。START 表示程序的起始, HALT 表示程序的终止。方框中是同时赋值语句,如 $(y_1, y_2) = (0, x_1)$, 表示将 y_1 置 0 值的

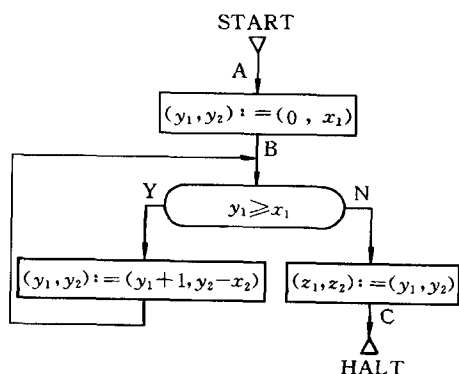


图 1 非负整数的除法框图

同时,将 y_2 的值置为 x_1 。圆框内是测试语句,用于控制程序加工的流程。如框图中的语句 $y_1 \geq x_1$ 表示当 y_1 的值大于等于 x_1 时,程序按 Y 的箭头继续执行,否则按 N 的箭头继续执行。

若约定各个变量的取值都是整数,上述除法程序的输入谓词和输出谓词分别为 $(x_1 \geq 0) \wedge (x_2 \geq 2)$ 和 $(x_1 = z_1 x_2 + z_2) \wedge (0 \leq z_2 < x_2)$ 。

在用归纳断言方式证明程序正确性时,还必须在程序的框图中设置一些数学公式,称作断言,表示程序执行到该处时,程序中变量应满足的数学关系。输入谓词可选作起点处的断言,而输出谓词可选作终止点处的断言。

在除法程序中设置三个断言, A 处和 C 处的断言分别为上述输入和输出谓词, B 处断言为

$$(x_1 = y_1 x_2 + y_2) \wedge (y_2 \geq 0) \quad (1)$$

反映了 y_1, y_2 中存放商数和余数的中间结果值。

验证程序的正确性,就是证明在程序的任何一种可能的加工过程中所设置的断言都是成立的。程序的一个加工过程就是框图中的一个流程。除法程序的所有可能的流程都是由图上的三条路径组合而成:由 A 至 B;由 B 出发回到 B;由 B 至 C。这样,验证程序的正确性,就是证明对任一条路径,只要起点的断言成立,则终止的断言也成立。

以第二条路线为例,它是一条环路。要证明下列命题:若程序执行到环路的起点 B 时,断言(1)成立,则程序执行一周,再达到 B 点时,断言(1)仍然成立。

环行该圈,就是在 $(y_2 \geq x_2)$ 成立的条件下,执行赋值语句 $(y_1, y_2) = (y_1 + 1, y_2 - x_2)$,而上述语句的执行结果是使 y_1 的取值为执行前 y_1 的值加 1, y_2 的取值为执行前 y_2 的值与 x_2 的差,其他变量的值不变。为保证执行该赋值语句后断言(1)仍然成立,就要求将断言(1)中的 y_1 代为 $(y_1 + 1)$, y_2 代为 $(y_2 - x_2)$ 后得到的公式在执行该语句前成立。即

$$(x_1 = (y_1 + 1)x_2 + (y_2 - x_2)) \wedge (y_2 - x_2 \geq 0) \quad (2)$$

在执行上述赋值语句前成立。但已知执行该语句前断言(1)和测试条件 $(y_2 \geq x_2)$ 均成立。由此推断公式(2)是成立的。这样就完成了对第二条路线的验证,对其余两条路线的验证类似,从而可以证明除法程序的正确性。

归纳断言方法由建立断言和对各条路径逐条验证两部分组成。建立断言是一种创造性的工作,而

验证路径的工作尽管繁琐,却是机械的。如何由计算机系统协助用户归纳出合适的断言,是程序验证程序中的重要课题。

用上述方法只能证明在输入谓词成立的前提下,程序终止时输出谓词一定成立。但不能证明在输入谓词成立时,程序一定能终止。不讨论程序终止性的程序验证称为程序部分正确性的验证。包括终止性的验证,则称为程序完全正确性的验证。

程序验证技术除了用于证明程序的正确性,或辅助用于编制正确程序外,还可从程序正确性角度评价程序设计方法和程序设计语言的优劣。但是,保证程序正确性的有效办法,不是在编制程序后再去验证,而是设法在编制过程中,使用适当的技术,使产生的程序正确无误。这类技术叫作程序综合和程序变换。程序验证技术和程序综合变换技术相互参照,共同发展。

参考文献

Manna Z. Mathematical Theory of Computation. McGraw-Hill, 1974 (李晓山 周巢尘)

chengxu zhuanhuan fangfa

程序转换方法 (program transformation method) 由一个程序转换至与之等价的另一个程序的方法。从软件自动化的角度看,程序转换有两类:一类是**纵向转换**,即由一抽象级别较高的程序转至另一满足功能要求的抽象级别较低的程序。另一类是**横向转换**,即在相同(或类似)抽象级别上程序间的转换。

纵向转换 涉及从功能规约到可执行的程序的全过程,可分为三个阶段:①从“做什么”的功能规约到“如何做”的设计规约转换;②从设计规约到高级语言程序的转换;③从高级语言程序到机器语言程序的转换。一般认为,①是其核心和难点,而②、③可借助编译技术完成。前者所要考虑的主要问题有:转换模型、转换过程中的正确性架构、知识及其表示机制和转换规则的选取等。代表性的工作是20世纪70年代中期西德慕尼黑技术大学信息学研究所在F. L. Bauer教授主持下开始研究的软件自动化系统,即计算机辅助、直觉指导的程序设计(CIP)项目,以及南京大学计算机软件研究所徐家福教授主持研制的算法设计自动化系统NDADAS和软件自动化系统NDAUTO。

CIP的目标是开发可形式保证程序正确性的程序开发系统。其课题有三:第一,设计并定义一广谱

语言CIP-L;第二,开发一交互系统;第三,建立一指导程序开发中形式推理过程的方法学。CIP的特点是:第一,各步之间转换的实现只借助“保证正确性”的转换规则;第二,开发全过程由程序人员指导,亦即,由程序人员选定转换规则。

NDADAS是实现功能规约到设计规约自动转换的算法设计自动化系统。它采用功能规约分解树模型,从给定的函数功能规约开始的转换过程由一系列自顶向下的精化步构成,每一精化步将某一函数规约或按某一控制结构分解成若干子函数功能规约;或将某一函数规约转换成另一易于求解的函数规约或算法已知的基元函数功能规约。此过程一直进行到所有未分解的(子)函数功能规约均是基元函数功能规约为止。结果即为相应的设计规约。

NDAUTO是实现设计规约到可执行的程序代码自动转换的软件自动化系统。主要工作是设计了一个图形化的软件设计规约语言GSPEC和实现了一个自动转换系统。GSPEC基于函数模型,并引入了新的软件分解模式,以及代数与一阶谓词相结合的抽象数据类型定义方法。系统转换过程分两步:第一,由转换程序将设计规约中的一阶谓词描述成分消去,将用户新定义的抽象数据类型用已有的类型实现;第二,由代码生成程序采用过程方法将转换后的设计规约翻译成可执行的程序代码。转换程序采用输入输出谓词综合法消去一阶谓词成分。

横向转换 代表性的工作是70年代初英国爱丁堡大学R. M. Burstall和J. Darlington的POP-2系统和在此基础上发展起来的ZAP系统。这类系统采用fold/unfold方法,根据一组基本规则自动地将作用式递归程序转换成功效较高的命令式循环程序,且只需少量的用户干预。

fold/unfold方法的转换对象是一阶递归方程语言。其转换规则有:定义、例化、展、卷、抽象及定律六条,主要作用是由已有方程生成新的方程。该方法的基本思想是,对所给方程通过展、抽象等规则进行展开,然后再通过卷规则对之进行一系列的重新合成,从而通过对程序正文的静态改变达到对程序动态计算功能的优化。基本策略是:①给出必要的定义;②引入置换实例进行例化;③依次“展”开;④尝试施用定律和where抽象;⑤依次“卷”起。其作用有二:第一,程序优化,它通过递归合并与递归化循环技术,将清晰的、易于理解的,但功效可能不高的程序转换成功效较高的程序。第二,程序综合,

可消除方程中的低效甚至是不可计算的逻辑成分和集合结构,而实现规约结构的转换,完全用递归方程加以刻画。

转换方法具有易实现、易修改和易扩充等特点,许多软件自动化系统采用了这一途径。其主要不足在于程序转换规则的自动选取十分困难,往往难以实现完全自动化。

参考文献

1. 徐家福,陈道蓄,吕建,王志坚. 软件自动化. 北京:清华大学出版社,1994
2. Partsch H A. Specifications and Transformation of Programs—a Formal Approach to Software Development. Springer-Verlage, 1990
3. Smith D R. Top-Down Synthesis of Divide-and-Conquer Algorithms. AI27(1):43~96

(王志坚 张家重)

chixu caiban he quanshouming zhichi

持续采办和全寿命支持(continuous acquisition and life-cycle support, CALS)

美国国防部联合工业界合作实施的一项大型网络应用工程计划。这个计划的实质是对于大型系统或产品实施全寿命信息管理,即在立项—竞标—研制—设计—生产—交付—培训—维护—报废—封存的全寿命过程中,有关各个方面携手合作,积极应用计算机和网络等先进信息技术,对系统或产品的技术信息数据(如图纸、三维造型、技术参数、规范标准、技术文档、技术手册等)按照标准进行数字化,逐步实现网络环境下的集成。不断改进工作过程和工作模式以适应信息环境,实现数据一次生成、多次传递使用,在全寿命周期内保持有标准、无误、通畅的产品技术信息数据流。提高信息数据的共享程度和可再利用性,从而提高工作效率,有效缩短研制生产周期、故障排除周期和改型换代周期,降低全寿命成本,提高产品质量和售后服务水平。

持续采办和全寿命支持(CALS)计划源于20世纪80年代中期美国国防部和工业界一项密切合作计划,该计划被称为计算机辅助后勤保障支持(缩写为CALS),旨在将武器装备技术手册和技术文档实施数字化。1987年又将此项计划修正为计算机辅助采办和后勤保障支持(依然缩写为CALS)。在不断总结实践经验的基础上,1993年,美国国防部将CALS计划正式命名为“持续采办和全寿命支持”。实践表明,CALS思想不但适合于军事工业,同样也适合于

电力、建筑、汽车、铁路、造船、民用航空等工业。美国商务部接受了CALS思想,并与电子商务联系起来,将其称之为光速商务(缩写仍为CALS)。

CALS全寿命信息管理的最终目标是营造一个数据集成环境(IDE)。在欧洲称之为数据共享环境(SDE)。推广实施CALS的前提是具备先进的信息基础设施,广泛应用计算机,并且制定发布一系列有关技术标准和法律法规。在此基础上,实施CALS过程主要做好三项工作:①推广交互式电子技术手册(IETM);②推动主承包商技术信息集成服务(CITIS);③重构工程化过程,以适应信息化数据集成环境。

许多国家的实践表明,实施CALS不但能有效地缩短研制、设计、生产(施工)和故障修复周期,降低全寿命成本,提高产品或工程质量,而且由于加强了信息数据的标准化、可交换性和共享性,推动了信息化国际接轨和营造了信息共享环境,因此有助于工业企业利用网络组织异地设计、生产和服务,参与国际合作和国际竞争,推动全球经济一体化发展。CALS也是发展工业信息化重要战略措施。

(赵孟林)

chongxie guize fa

重写规则法(rewriting rule method)

使用重写规则的定理机器证明方法。它是D. E. Knuth和P. B. Bendix于1970年提出来的。重写规则法的最初目标是解决等式中的推理问题:设有等式集合 $E = \{t_1 = u_1, t_2 = u_2, \dots, t_n = u_n\}$,问等式 $t = u$ 是否能从等式集合 E 通过等价公理和适当的替换推导出来。Knuth和Bendix把集合 E 中的等式看成是只能从左向右使用的重写规则,把对项的推导过程看做是使用重写规则的简化过程。他们经过深入研究后指出:可以对某些重写规则集增加一些相容的重写规则,使之成为标准重写规则集。对于标准重写规则集,每一个项 t 都有惟一的不能再化简的最简形式 t^* 。于是,在标准重写规则集下,要判断两个项 t 和 u 是否相等,只要看它们的最简形式 t^* 和 u^* 是否相等就可以了。他们还建立了著名的构造标准重写规则集的Knuth-Bendix算法。他们使用重写规则法成功地解决了群论中等式的证明问题。

虽然重写规则法是就等式推理提出的,但是人们发现,许多其他类型的推理问题经过适当地处理之后,也可以用重写规则法来解决,因此引起很多人的兴趣,纷纷对它加以研究和改进,使其应用领域日

益广泛。20 世纪 70 年代末, D. S. Lankford 等人提出了交换结合合一的概念(AC 合一), 把运算的交换律和结合律隐含在合一算法中, 不作为规则使用, 成功地解决了 Knuth-Bendix 算法对含有交换律的系统不适用的问题。他们还建立了布尔代数标准重写系统, 实际上已经解决了重写规则法在命题逻辑中的应用问题。80 年代初, J. Hsiang 等人以布尔环为工具, 利用布尔环与布尔代数的等价关系, 解决了布尔代数的表示不惟一问题, 不仅成功地把重写规则法用在命题逻辑上, 而且进一步推广到一阶逻辑上, 建立了使用重写规则的定理证明系统。实验表明, 使用这种证明系统所产生的新项数比使用归结法所产生的归结式数或是使用自然演绎法所产生的子目标数都要少。

重写规则法在定理证明过程中不需要折回, 搜索空间比较小。在建立了标准重写规则集之后, 证明算法还是完备的, 因此近年来成为定理证明中一种引人注意的新方法。

参考文献

1. Knuth D E, Bendix P B. Simple Word Problem in Universal Algebras. In: Leach J, ed. Computational Algebra. New York: Pergamon Press, 1970, 263 ~ 297
2. Lankford D S, Ballantyne A M. Decision Procedure for Simple Equational Theory with Commutative-associative Axiom. Rept. ATP-39, University of Texas at Austin, TX, 1977 (姜云飞)

chouxian daishu

抽象代数 (abstract algebra) 在初等代数的基础上, 通过对数系概念的发展, 而于 20 世纪 20 年代建立起来的代数学分支。又称近世代数。

初等代数学是研究实数或复数和以它们为系数的多项式的代数运算(加法、减法、乘法、除法、乘方和开方等)的理论和方法, 其研究方法是高度计算性的, 其中心问题是代数方程和代数方程组的解的求法及其分布的研究。抽象代数学是以研究数字、文字和更一般元素的代数运算的规律和由这些运算适合的公理所定义的各种代数结构(或称代数系统)的性质为其中心问题, 其研究方法主要是公理化的。从各种代数结构的公理出发研究它们的性质, 就是所谓的抽象代数学。现在已有群、环、域、模、代数、格、同调代数、范畴和泛代数等一些重要代数结构。所谓代数结构是由一个集合和定义在这个集合中的一种或若干种运算所构成的

一个系统。

模 设 R 为一个环, A 为一个交换群。若 R 中任意元素 r 和 A 中任意元素 a 都可结合为一个元素, 用 ra 表示, 并且还满足以下公理:

$$M_1 \quad r(a+b) = ra + rb$$

$$M_2 \quad (r+s)a = ra + sa \quad r, s \in R \text{ 且 } a, b \in A$$

$$M_3 \quad r(sa) = (rs)a$$

则称 A 为(左) R -模。若 R 具有单位元 I_R 且还满足公理

$$M_4 \quad I_R a = a \quad a \in A$$

则称 A 为么作用(左) R -模。当 R 为体时, 称么作用(左) R -模 A 为(左)向量空间。

可类似地引进右 R -模、么作用右 R -模和右向量空间。

代数 设 K 为一个具有单位元的交换环, A 为么作用 K -模。如果对任意 $k \in K$ 和 $a, b \in A$, 有 $k(ab) = (ka)b = a(kb)$, 则称 A 为一个 K -代数。若 K -代数 A 是体, 则称 A 为可除代数。

设 F 为一个域且 n 为正整数。若对任意 $c \in F$ 和 F 上任意 n 阶方阵 $A = (a_{ij})$ 和 $B = (b_{ij})$, 定义

$$A+B = (a_{ij} + b_{ij}), A \cdot B = \left(\sum_{k=1}^n a_{ik} b_{kj} \right), cA = (ca_{ij})$$

则所有 F 上 n 阶方阵的集合 $M_n(F)$ 就是一个 F -代数, 称为 F 上的 n 阶矩阵代数。

泛代数 如果 F 为函数符号的非空集合, σ 为从 F 到自然数(包括 0)的映射, 则称 $\Omega = \langle F, \sigma \rangle$ 为一个型。若 $f \in F$ 使 $\sigma(f) = k$, 则称 f 为 Ω 的一个 k 元函数符号。一个 Ω -泛代数(简称泛代数)就是一个二元偶 $U = \langle A, \Omega \rangle$, 其中 A 为一个非空集合, 称为 U 的载体, 并且使 Ω 的每个 k 元函数符号 f 都对应于 A 上惟一的 k 元运算 $f^U: A^k \rightarrow A$ 。

显然, 群、环、域、格和布尔代数等都是泛代数。

型 $\Omega_1 = \langle F_1, \sigma_1 \rangle$ 与 $\Omega_2 = \langle F_2, \sigma_2 \rangle$ 称为等同的, 是指有双射 $\rho: F_1 \rightarrow F_2$ 使 $\sigma_1 = \sigma_2 \circ \rho$ 。泛代数 $U_1 = \langle A_1, \Omega_1 \rangle$ 与 $U_2 = \langle A_2, \Omega_2 \rangle$ 称为同型的, 是指 Ω_1 与 Ω_2 是等同的。可以由同型泛代数构造积泛代数, 并可在同型泛代数之间建立同态与同构的概念, 而且相应的同态定理成立。

设 $U = \langle A, \Omega \rangle$ 为一个泛代数且 \approx 为 A 上一个等价关系。若对每个 Ω 的 k 元函数符号 f , 当 $a_i, b_i \in A$ 使 $a_i \approx b_i (i=1, 2, \dots, k)$ 时恒有 $f^U(a_1, a_2, \dots, a_k) \approx f^U(b_1, b_2, \dots, b_k)$, 则称 \approx 为 U 上一个同余。这时, 若把每个 $a \in A$ 所在的等价类记为 \bar{a} , 所有等价类的集合记为 \bar{A} , 并对每个 Ω 的 k 元函数符号 f 都定义 \bar{A}

上一个 k 元运算 $f^U: \bar{A}^k \rightarrow A$ 如下:

$$f^U(\bar{a}_1, \bar{a}_2, \dots, \bar{a}_k) = f^U(a_1, a_2, \dots, a_k) \\ a_1, a_2, \dots, a_k \in A$$

则可获得一个新 Ω -泛代数 $\bar{U} = \langle \bar{A}, \Omega \rangle$, 称为 U 关于 \approx 的商泛代数, 记为 U/\approx 。

由于代数结构及其元素的极其一般性, 特别是代数运算贯穿于任何数学理论和应用问题之中, 所以抽象代数早已渗透到各个不同的数学领域中, 并形成了一些新的数学领域, 诸如代数数论、代数几何、拓扑代数、代数拓扑、李群、李代数、泛函分析等。抽象代数不仅对全部现代数学有显著的相互影响, 而且对其他科学领域如计算机科学、理论物理和结晶学等也都有重要影响。特别是随着计算机科学和技术的迅速发展, 抽象代数的一些成果和方法更被直接应用到计算机科学和某些工程技术中, 并与计算机科学的某些分支相互影响、相互结合而产生了一些具有新面貌和新风格的研究领域, 诸如基调代数、逻辑代数、代数编码学、进程代数、语言代数学、代数语言学、代数自动机理论、代数语义学等。

参考文献

1. Jacobson N. Basic Algebra. Vol 1, 2. Freeman, San Francisco, 1974, 1980
2. 王兵山, 周贤林, 王长英编. 离散数学. 长沙: 国防科技大学出版社, 1985
3. Burris S and Sankappanavar H. A Course in Universal Algebra. New York: Springer-Verlag, 1981

(李廉 王兵山)

chouxian shuju leixing

抽象数据类型 (abstract data type) 与表示无关的数据类型。数据类型由一个对象集合 (值集) 和在该集合上定义的若干合法运算所组成的运算集合组成。抽象数据类型用数学方法定义对象集合和运算集合, 仅通过运算的性质刻画数据对象, 而独立于计算机中可能的表示方法。其目的在于隐蔽运算实现细节和内部数据结构, 同时向用户提供该数据类型的完整信息。

抽象数据类型的概念是逐步形成的。20 世纪 60 年代末到 70 年代, 人们将用户自定义的类型称作抽象数据类型。传统的算法语言没有用户自定义类型的设施。到 60 年代末, 为了实现算法细节和数据内部结构的隐蔽, SIMULA 67 语言中引入了类, 随后出现了模块概念。模块可分成模块式和模块体, 模块式定义外部可见的运算接口, 模块体对外不可

见, 其中可定义私有的数据结构和运算。通过接口和实现的分离, 模块提供了用户自定义类型的手段, 达到数据抽象、信息隐蔽的目的。在这个意义下, 模块所定义的数据类型称作抽象数据类型。实际上, 模块提供了一种抽象数据类型实现的手段。这种手段在 Modula-2, Ada 等语言中得到进一步的完善和发展。用户自定义数据类型的另一优点是设计与实现相分离, 可将模块式看作设计规约, 而模块体是相应的实现。这种分离推动了软件规约的研究, 也进一步推动了抽象数据类型研究。

当用一个数据类型去模拟一类客观对象时, 可先给出该类型的性质和功能的描述, 然后用已有的语言设施和数据类型实现所需的功能, 并证明实现的正确性。仅通过模块式描述数据类型的型构是不够的, 还必须用抽象的方法完整地描述对象的性态和功能。这就是用抽象数据类型表示功能规约。这时, 抽象数据类型完全独立于具体表示, 反映出纯抽象的性质。抽象数据类型的规约方法主要有二: 其一是代数方法; 其二是模型方法。代数方法基于 G. Birkhoff, J. D. Lipson 等的异调代数理论, 经 S. Zilles, J. A. Gutttag 等人的发展, 其理论基础日趋完善, 并逐步应用于软件工程实践, 成为有代表性的抽象数据类型规约方法。模型方法基于 C. A. R. Hoare 的**前后断言方法**, 它通过已定义的 (抽象) 数据类型来给出所要定义的新类型的抽象模型。

采用代数方法, 抽象数据类型的规约由两部分组成, 一是语法部分, 二是公理部分。语法部分给出了抽象数据类型的名及其上运算的定义域和值域, 公理部分则通过给出一组刻画各运算之间相互关系的方程来定义各运算的含义。从语义的角度, 代数规约的语义是一类代数。在语法正确的基础上, 语义正确性是指相应代数满足规约中公理部分的所有公理。通常, 具语义正确性的语义模型代数有多个且形成一个谱, 谱的两端分别称为始语义模型和终语义模型。该谱系为实现者提供了较大的灵活性。基于代数方法的规约语言有 OBJ, Clear 等。例如用 OBJ 写的栈定义如下:

```
obj stack;
sort stack /integer, boolean;
ok_ops

push: stack, integer → stack;
pop:  stack → stack;
top:  stack → integer;
empty: stack → boolean;
```

```

newstack: → stack;
depth: stack → integer; hidden;
error-ops
underflow → stack;
no_more → integer;
overflow → stack;
op-equ's
pop( push( s, item ) ) = s;
top( push( s, item ) ) = item;
empty( newstack ) = true;
empty( push( s, item ) ) = false;
depth( newstack ) = 0;
depth( push( s, item ) ) = 1 + depth( s );
error-equ's
pop( newstack ) = underflow;
top( newstack ) = no_more;
push( s, item ) = overflow if depth( s )
> 100;
job

```

模型方法不是对抽象数据类型中运算的性质加以直接刻画,而是通过某些基本类型和已定义的(抽象)数据类型来给出所要定义的新类型的抽象模型,用已定义的运算的性质来间接刻画要定义的数据类型中运算的特性。一般说来,抽象数据类型的模型规约由以下三部分组成:①状态集的定义(可能包含不变式);②初始状态定义(通常只有一个);③运算集的定义,通常采用输入输出断言刻画。值得注意的是,模型只能理解成行为的描述而与具体实现无关。但如果不加注意,模型规约可能引入与实现有关的内容。由于这种方法与正确性证明方法有较直接的对应,因此,在规约语言中得到了广泛的应用。如Z, VDM等。

抽象数据类型能够从抽象的角度描述客观对象的性态,满足信息隐蔽、功能抽象、设计在前、方便验证等软件工程的要求。其理论对软件规约、正确性证明等课题的研究均具有重要意义,对面向对象语言的发展产生了重要的影响。

参考文献

1. Liskov B H, Gutttag J. Abstraction and Specification in Program Development. Cambridge: The MIT Press, 1986

2. Gutttag J V. Notes on Type Abstraction. In: Proc. Specification of Reliable Software Conf., Cambridge, 1979 (伊波 吕建)

chufaqi

除法器 (divider) 对以数字形式表示的两个 n 位数求商的一种运算电路。

两个原码数相除,其商的数值为两数绝对值相除后的结果,而符号为两数符号的异或值。设 x, y 为被除数与除数, x_s, y_s 为被除数与除数的符号,则其商 $Q = |x|/|y|$, 符号 $Q_s = x_s \oplus y_s$ 。

除法运算虽然使用频度低,但它是基本的四则运算之一,因此大多数计算机有除法器。与乘法的处理思想相似,除法运算的常规算法是将除法转换成若干次“加减-移位”循环。常见的除法算法有恢复余数法和恢复余数法。恢复余数法是,不管被除数(或余数)减除数是否够减,都一律做减法。若余数为正,表示够减,该位商上“1”;若余数为负,表示不够减,该位商上“0”,并要恢复原来的被除数(或余数)。恢复余数法一般很少采用,原因是恢复余数时浪费一次加法时间,同时造成除法进行过程的步数不固定,控制起来比较复杂。最简单并被广泛采用的除法器是不恢复余数法除法器。

二进制原码不恢复余数除法的规则可由下式表示:

$$r_{i+1} = 2r_i + (1 - 2Q_i) \times y$$

式中, r_i 和 r_{i+1} 分别为第 i 次和第 $i+1$ 次求商所得的余数, Q_i 为第 i 步所得的商,即当余数为正时,上商为“1”,余数左移 1 位,下一步减除数;当余数为负时,上商为“0”,余数左移 1 位,下一步加除数。由于加减运算交替进行,故也称为原码加减交替法。

现举例说明原码加减交替除法的实现。设 $x = 0.10101, y = 0.11110$, 则

$$\frac{x}{y} = \frac{0.10101}{0.11110}$$

$|x| = 0.10101 \rightarrow A(\text{寄存器}), |y| = 0.11110 \rightarrow B(\text{寄存器})$

$[-B]_{\text{补}} = 1.00010, 0 \rightarrow C(\text{寄存器})$

运算过程如下:

A	C	
00.10101	0.00000	
+ [-B] _# 11.00010		- y
11.10111	0.00000	余数为负, 商0
← 11.01110		左移一位
+ B 00.11110		+ y
00.01100	0.00001	余数为正, 商1
00.11000		左移一位
+ [-B] _# 11.00010	0.00010	- y
11.11010		余数为负, 商0
← 11.10100		左移一位
+ B 00.11110	0.00101	+ y
00.10010		余数为正, 商1
← 01.00100		左移一位
+ [-B] _# 11.00010	0.01011	- y
00.00110		余数为正, 商1
← 00.01100		左移一位
+ [-B] _# 11.00010	0.10110	- y
11.01110		余数为负, 商0
+ B 00.11110		
00.01100		最后一步恢复余数, + y

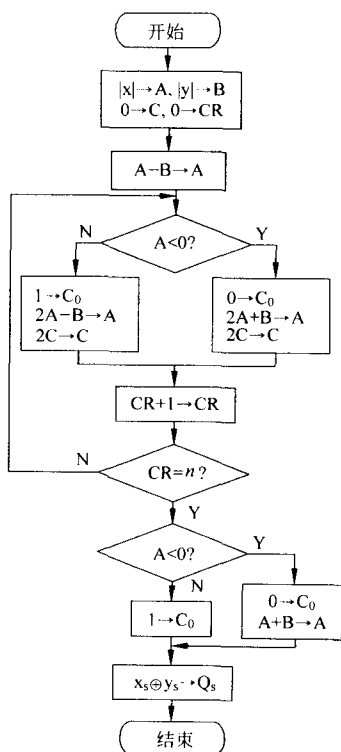


图1 原码加减交替除法流程

又由 $Q_s = x_s \oplus y_s = 0 \oplus 0 = 0$, 得 $\frac{x}{y} = 0.10110 + \frac{0.01100 \times 2^{-5}}{0.11110}$ 。

实现原码加减交替除法器的算法流程如图1所示,其中三个寄存器 A,B,C 分别存放被除数、除数和商,计数器 CR 用来控制移位次数, C_0 为 C 寄存器的最低位。

需要注意的是,在定点小数除法运算时,为了防止溢出,要求被除数的绝对值小于除数的绝对值,且除数不能为0。另外,在原码加减交替法中,当最终余数为负数时,必须恢复余数,使之变为真余数,但此时不能再左移了。

为了提高除法的运算速度,可采用跳0跳1除法、阵列除法器或迭代除法。跳0跳1除法的原理是:当前一步除法所得余数很小,即可连商多位,余数前面的0越多,连商的位数就越多,这将减少除法的步数,从而提高除法的速度。阵列除法器是利用若干个加减单元组成阵列,将各步“加减-移位”操作在一步内完成。迭代除法的基本思想是通过多次乘法运算来获得商,以加快求商的速度。

参考文献

1. 蒋本珊. 电子计算机组成原理. 北京: 北京理工大学出版社, 1994

2. 王爱英主编. 计算机组成与结构(第三版). 北京:清华大学出版社, 2001 (刘恩德)

chuliji tixi jiegou

处理机体系结构 (processor architecture)

从汇编语言程序设计员的角度所看到的处理机的内部功能结构。

计算机体系结构可分为两个层次:一是处理机级的;二是计算机系统级的。后者不仅包括处理机,还包括存储器与外围设备。处理机体系结构一般包含计算机的运算器、控制器以及汇编语言程序员所用到的寄存器。

处理机体系结构的基本组成

处理机体系结构是围绕着计算机指令系统的执行来设计的。一条典型指令的执行包括以下阶段:取指令、指令译码、计算存放操作数的有效地址、取操作数、执行指令所规定的操作以及送回操作结果。不同的指令在各个阶段的操作不同,控制指令的操作是处理机中控制器的职能。执行指令的算术运算或逻辑操作的部件是处理机的运算器。在处理机中暂时存放指令、控制字、源操作数、中间结果和最后结果的是处理机中的寄存器。控制器、运算器和寄存器是构成处理机体系结构的基本部件。新型的处理机体系结构还包括流水线执行机制、指令发射与调度机制、片上高速缓冲存储器、存储管理机制、协处理器和转移预测机制等部件。

几种典型的处理机体系结构

(1) 微程序控制器和复杂指令集计算机 在计算机发展初期,处理机采用**硬连线控制器**。自从 20 世纪 60 年代中期美国 IBM 公司推出 IBM 360 系列机以来,要求处理机体系结构能满足软件的向上兼容性,即在低档机上开发的软件应在同一系列中的高档机中二进制兼容,作为控制指令执行的核心——控制器必须满足这一要求。此外,由于当时的中央处理机已采用双极型半导体器件,工作速度较快,而存储器仍然以磁心为基础,存取周期较长,为中央处理机工作周期的几倍(一般为 5 倍左右)。这样,以微程序技术为基础的控制(参见**微程序控制器**)是很合理的选择,它曾经在处理机中沿用了二十多年。微程序控制器中的微存储器也采用了较快的器件,如磁膜和半导体器件。因其容量比主存储器的容量小得多,采用较昂贵但快速的器件是合算的。这种微存储器的周期与中央处理机的工作节拍相吻合,允许中央处理机 1 拍执行 1 条微指令。

主存储器周期(即访问主存储器所需的时间)等于若干个(5 到 8 个)微周期,而 1 条指令的执行也正好由若干条微指令组成。因此,在当时的计算机工艺条件下,处理机的控制器采用微程序技术是和工艺条件相匹配的。同时,微程序技术又容易满足软件向上兼容性的要求。微存储器是控制部件的“核”,如果在同一系列中,要设计指令系统功能更强的高档机,只要把这个“核”扩充就可以了。因为复杂的指令可以分解成很多个基本的微指令,从而容易使系列产品中低档机与高档机的软件二进制兼容。

由于微程序技术便于实现复杂的指令系统,20 世纪 60 年代以后的传统计算机的指令系统日趋复杂,称为**复杂指令集计算机(CISC)**。这种计算机的处理机的特点是:指令系统庞大;指令种类和寻址方式较多,指令多采用存储器-存储器操作以及存储器-寄存器操作;控制器绝大部分采用微程序技术,一条指令的执行时间包含多个乃至几十个微周期。指令字长是可变的,从两个字节到十几个字节。

(2) 精简指令集计算机 **精简指令集计算机(RISC)**的处理机体系结构在 20 世纪 70 年代后期形成设计思想,80 年代中期发展成产品,并获得迅速发展。后来几乎所有的超大规模集成电路(VLSI)处理机都采用 RISC 技术,或者正在向 RISC 过渡。RISC 处理机体系结构的主要特点是:在指令系统中尽量采用在程序设计中频繁使用的那些基本指令;尽量采用寄存器-寄存器指令类型;把存储器访问指令减少到最少程度,只剩下不可缺少的 LOAD/STORE 指令,故 RISC 结构又称为 LOAD/STORE 结构。处理机主要采用硬连线控制器。在 RISC 处理机中,可做到 1 条基本指令在 1 个周期内执行完毕,指令周期相当于 CISC 处理机中的 1 个微周期。RISC 处理机中指令长度是固定的,指令操作码字段和操作数地址字段等都放在固定位置,从而使译码器电路简化和译码时间缩短;指令类型和寻址方式种类也尽量少。

RISC 处理机的主要优点在于其指令系统内的多数指令可在 1 个机器周期内执行完毕,但这一点必须在流水线结构(参见**计算机流水线**)的基础上完成。RISC 处理器采用流水线结构后,会产生数据相关和转移相关问题。为了使流水线执行效率提高,使执行每条指令的平均周期数(CPI)减少,必须采用编译优化技术中的指令调度来尽可能减少上述

相关性的影响。因此,RISC 技术和流水线技术以及编译优化技术密切相关。RISC 体系结构不但提高了处理机的处理速度(从 1987 年到 1994 年,RISC 处理器的处理速度从 3 MIPS 提高到了 300 MIPS),目前单个通用处理机的峰值速度已达到几十亿条指令每秒。处理机性能的迅速提高,主要是由微电子工艺技术和处理机体系结构技术这两方面的发展所促进的。处理机性能的迅速提高已对计算机产业的发展产生了深刻的影响,它改变了微型计算机、小型计算机和大型计算机的产品结构,促进了计算机应用的飞速发展。

(3) 多发射结构和指令级并行处理 在 RISC 结构中,如果每个周期只发出 1 条指令,则无论流水线设计如何精细,编译优化指令调度技术如何发挥,RISC 处理机的 CPI 不可能小于 1。或者说,每个周期平均执行的指令数(IPC)不可能大于 1。每周期只发出 1 条指令的 RISC 结构称为单发射 RISC 结构;每个周期发出多条指令的 RISC 结构称为多发射 RISC 结构(参见多发射结构)。20 世纪 90 年代以来,几乎所有 RISC 处理机都采用多发射结构。在多发射结构中,在每一个机器周期内发射多条指令,这固然可以提高 IPC,但也带来新问题,即在每个周期内,这多条指令之间可能有数据相关或转移相关。同时,在流水线结构中,1 个周期内取出的多条指令与其后的周期内取出的多条指令之间又可能发生数据相关或转移相关。如何避免多发射结构中复杂的相关,加强指令调度技术以取得更高的指令执行并行性,从而提高 IPC 与计算机性能,就成为处理机体系结构设计的重要课题。在新型处理机体系结构中,硬件与软件(包括编译与操作系统)是不可分割的。为了解决相关,进一步提高 IPC 与指令执行并行性,必须采用硬件与软件相结合的综合措施,这一技术称为指令级并行处理(ILPP)。ILPP 技术的作用在于可以提高计算机的性能,同时完全保持软件的兼容性,因为它对用户是透明的。而传统的并行处理技术虽然可以提高计算机性能,但必须对应用程序进行划分,或采用并行编译,从而较难做到应用软件的兼容。另外,还要求用户具有专门的技术,方可发挥并行处理系统的效率。多发射结构中的指令并行执行还有一个流水线的资源冲突问题。为了减少流水线资源冲突,较新的处理机体系结构,例如 Intel 公司的 Pentium 系列都采用了多流水线结构,如具有两条执行整数指令的流水线与 1 条执行浮点数指令的流水线。1 个周期中发射的 3 条指令有可

能在各自的流水线中执行而不发生资源冲突。

在多发射处理机体系结构中,有人提出“超长指令字处理机(VLIW)”结构,即每个指令字可以包括 16 条或 32 条指令,同时发射。然后,依靠一种称为“跟踪-调度”的编译优化技术来达到尽可能多的指令并发地执行。但是,实践证明:这种编译技术在 VLIW 处理机编译后的执行代码长度膨胀过大,以至于执行速度提不上去。不过,这种 VLIW 的思想被部分接受,称为长指令处理机(LIW),每个指令字包含 4 条或 8 条指令。LIW 已用在 Pentium III 和 IV 中。

以上是处理机体系结构从 20 世纪 60 年代以来的发展主流。在发展过程中,还有两种体系结构值得一提。一种体系结构是中央处理机加上协处理器。在这种系统中,常规的处理由中央处理机完成,而一些特殊的处理由协处理器完成,这样,中央处理机可不致太复杂,协处理器由于专门从事某种处理而效率较高。例如,专门处理浮点数操作的协处理器、专门处理图形的协处理器、专门处理数字信号的协处理器以及专门处理知识推理的协处理器等。最近,随着因特网的发展,在很多处理机芯片中,实际上加入了处理信息压缩的或处理因特网上静态与动态视频图像的 MPEG 的协处理器。这些协处理器和中央处理机共同监视正在执行的指令流,以便从中分出自己应执行的指令或任务。此外,它们之间还要有通信接口与同步机制。这些协处理器可以是和中央处理机分开制作的,也可以做在同一芯片上,如 Intel 80860 芯片包含了中央处理机、浮点数协处理器以及图形协处理器。另一种体系结构为高级语言处理机结构。这种处理机可分为直接执行高级语言处理机(如执行 LISP 语言的 LISP 处理机和执行 Forth 语言的 Forth 处理机)和面向高级语言、依靠微程序来间接执行高级语言的处理机(如面向 Ada 语言的 Intel iAPX 432 处理机)。但这些高级语言处理机都很不成功。因为常规的处理机已经大批量生产,价格较便宜,而且已有大量应用软件和被普遍接受的操作系统的支持。此外,高级语言处理机仍大量采用存储器访问类型的指令,花费大量时间在存储访问上,因而它们执行高级语言的实际速度反而比经过编译优化的 RISC 处理机慢很多。

处理机体系结构的发展方向

处理机体系结构的发展方向是:提高指令执行的并行度和流水线执行效率。除了增加每周期发射的指令数量,优化指令调度以外,还要增设执行指令

操作的执行部件,减少指令执行时的资源冲突;同时提高存取操作数的速度和并行性。因此需要发展多端口的寄存器堆和片上高速缓冲存储器。为了减少转移相关,较新型的处理机为加强转移预测和转移处理的能力,在处理机芯片中设计了各种转移预测和转移处理部件。

一个值得注意的新型处理机体系结构是多线程处理机结构。这种结构在指令发射以前,先对指令进行整理,将指令流分解成若干个不相关的指令流线程,然后分别发射到不同的流水线中去执行。

多处理机计算机系统是计算机发展的一个重要方向,因此在处理机体系结构中,也会增强支持多处理机的能力,如支持高速缓存一致性的措施和协议以及支持高速内部总线的能力等。在单一芯片上,集成多个处理机,也在积极研究中。

此外,由于现在计算机系统中,中央处理机访问其芯片外存储器数据的速度远远低于在同一芯片上数据访问的速度;这种速度的差距已成为计算机系统无法充分发挥处理机性能的主要障碍之一。因此,过去传统的、以中央处理机为主的系统结构观点受到了挑战。现在,很多单位在研究面向存储器的处理机体系结构,例如“存储器中的处理机(PIM)体系结构”,以有效增加处理机与存储器之间的数据传输的带宽。

参考文献

1. 李三立. 微处理器与微计算机. 北京:国防工业出版社,1981
2. Hennessy J L, Patterson D A. Computer Architecture: A Quantitative Approach. Second Edition. Morgan Kaufman Publishers Inc., 1996
3. 李三立,李亚民. RISC——单发射与多发射体系结构. 北京:清华大学出版社,1994 (李三立)

chuliqu guanli chengxu

处理器管理程序 (processor manager) 参见任务调度程序。

chuping

触屏 (touch screen) 一种用手指或笔触及屏幕上所显示的选项来完成指定的工作的人机交互输入设备。它使用方便,反应灵敏,不要求使用者有多少计算机知识,也不需要使用者记住太多的使用规则。因此,在许多场合得到广泛应用。

触屏在安装方式上有两种:一种是外挂式,把触屏附装在荧光屏外面;另一种是内装式,把触屏在生产显示器时装配成一体。

触屏由三个部分组成。一是传感器,把人的手指或笔触及的位置检测出来。二是控制卡,触及信号经过模数转换器,形成位置数据,经接口送入计算机。三是驱动程序,即相应的管理软件。

传感器有电阻膜、红外线、表面声波、电容和压力等五种,常用的是电阻膜和红外线两种。

电阻膜传感器是由两层透明的金属薄膜组成。当薄膜上某一点受到压力时,两层金属薄膜的这一点接触,造成短路,由此测出受压点的坐标值。电阻膜传感器的缺点是透光性差,只有 70% ~ 80% 的透光度,因而影响了显示器屏幕的亮度。它的优点是响应速度快,分辨率高(可达 4 096 点 × 4 096 点),而且价廉。

红外线传感器是在屏幕的上下和左右一一对应地装上红外发光管和红外光敏元件。当中间有手指或笔隔断光线时,相应的光敏元件就失去光信号,因此很容易检测出触及点的坐标。红外线传感器透光性好,因为它安装在显示器的边框上,不遮挡荧光屏。但它的分辨率不高,反应速度也慢。

触屏与主机连接多采用 RS-232C 接口。

触屏在个人计算机中使用不多,因为长时间举手,会感到劳累。但它在商业、金融等方面应用广泛。例如餐馆里点菜,百货公司里选购商品,旅游景点的介绍,交通信息的查询等,用触屏非常方便。

(林兼)

chuanshu kongzhi xieyi

传输控制协议 (transmission control protocol, TCP) Internet 定义两台计算机之间进行可靠传输、交换数据的面向连接的协议。协议定义了交换的数据和确认信息的格式,提出了为确保数据的正确到达而采取的措施。协议规定了怎样识别给定计算机上的多个目的的进程,如何对分组丢失和分组重复这类差错进行恢复。TCP 将数据流看作字节序列,为了便于传输又将这个序列划分为若干段。加上 TCP 头后交给 IP 协议发送。协议规定了两台计算机如何初始化一个 TCP 数据流传输以及如何结束这一传输。协议还规定了软件应提供的操作,但并没有指定应用程序调用的具体过程。

TCP 在协议层次结构中位于 IP (参见网际协

议)层之上。TCP 允许一台计算机上的多个应用程序同时进行通信,也能对接收到的数据进行分解,分别送到多个应用程序。TCP 使用协议端口号来标识一台计算机上的多个目的进程。每个端口都被赋予一个小的整数作为端口号以便识别。

由于 TCP 是建立在连接的抽象概念上的,它所标识的对象不是某个端口,而是一个虚电路连接。TCP 使用连接而不是协议端口作为基本的抽象概念,连接是用一对端点来标识的。

TCP 将端点定义为一对整数(host, port),其中 host 是主机的 IP 地址, port 是该主机上的 TCP 端口号。由于 TCP 使用端点来识别连接,这样一台计算机上的某个 TCP 端口号可以被多个连接所共享。因此,程序员能设计同时为多个连接服务的程序,而不需要为每个连接设置各自的本地端口号。

传输控制协议是一个面向连接的协议。它需要两个端点都同意连接才能进行通信。在以网络通信之前,连接双方的应用程序必须先建立连接。这种连接采用**客户-服务器模式**建立,即由客户方的应用程序主动打开请求,通知**网络操作系统**要建立一个连接,服务方应用程序通过操作系统得知客户方的请求后又通知操作系统,同意建立一个连接,对于服务方的应用程序来说,这个过程是被动打开的过程。连接建立之后,应用程序开始传输数据。

TCP 使用专门的滑动窗口机制来解决传输效率和流量控制这两个问题。TCP 允许随时改变窗口大小。在每个确认中,除了指出已经收到的分组外,还包括了一个窗口通告,用来说明接收方还能再接收多少数据。可以将窗口通告的值当做当前的接收缓冲区分大小。通告值增加,发送方扩大发送窗口;通告值减少,发送方缩小发送窗口。采用这种滑动窗口机制,不仅提供了可靠传输服务,而且还提供了流量控制功能。但是 TCP 采用的滑动窗口机制只解决了端到端的流量控制,并未解决整个网络的拥塞控制。

两台计算机上的 TCP 软件之间传输的数据单元称报文段。通过报文段的交互可建立连接、传输数据、发出确认、通告窗口大小以及关闭连接。

TCP 使用三次握手协议来建立和关闭连接。三次握手协议是连接两端正确同步的充要条件。因为 TCP 建立在不可靠的分组传输服务之上,报文可能丢失、延迟、重复和乱序,因此协议必须使用超时和重传机制。如果重传的连接请求和原先的连接请求在连接正在建立时到达,或者在一个连接已经建立、

使用和结束之后,某个延迟的连接请求才到达,就会出现問題。采用三次握手协议可以解决这些问题。TCP 通过交换包含最少量信息的报文段来实现三次握手。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999
2. Douglas E Comer. Internetworking with TCP/IP, Volume 1, 3rd ed. Prentice Hall Inc., 1995

(胡道元)

chuanshu sunhao

传输损耗 (transmission impairments) 由外部信号源或传输系统本身引起的信号畸变、衰减或失真。很多电气干扰都会使**传输介质**产生噪声,从而引起信号的畸变。例如,在无线电传输中,噪声常常会使扬声器输出产生嘘声或噼啪声。在电视机中,电视传输的噪声会使雪花形干扰或五彩纸屑形干扰重叠在图像上。在数字通信系统中,噪声可能会产生不希望有的脉冲,甚至会抵消所需要的脉冲,从而使所接收的信号产生严重的误差。此外,每一种电路都有电阻、电感和电容,当信号在电路中传输时必然会失真。上述各种失真都可认为是传输损耗。

传输损耗分类

可以将传输损耗分成两大类,即外部损耗和内部损耗。外部损耗是指外部信号源对传输系统产生的损害;内部损耗是指传输系统本身内部引起的损耗。

有三种不同类型的外部损耗:

(1) 静态噪声 大多数外部噪声(电气干扰)以随机无线电波形式存在,一般叫静态噪声。

(2) 地球外噪声 这种噪声在一个很宽的频谱上辐射,其中包括通信用的频率。地球外噪声又分太阳射电噪声和宇宙射电噪声两类。太阳射电噪声是由太阳的恒定辐射产生的;宇宙射电噪声是宇宙空间不同恒星产生的太阳射电噪声。

(3) 工业噪声 如飞机和汽车点火装置、电动机、开关设备,以及高压线的泄漏产生的噪声。荧光灯是这类噪声的另一个强大的干扰源,因此,不应使用于高灵敏接收机接收或测试的场所。

内部损耗也有三种不同类型:

(1) 衰减失真 信号的强度会随着它在任何一种传输介质上传输的距离增加而下降,这种现象称

衰减。衰减也称幅度衰变,它是话音(言语)级电话线路的一种固有特性。

(2) 延迟失真 数字信号是由很多频率成分组成的。当信号在通信介质传输时,所有被传输的频率成分应该在同一时间内以相同的电平接收到。实际上所有的频率成分并不是按相同的速率传输的,较高的频率成分传输的速度较低,而较低的频率成分传输的速度较高。由于这一原因,使某些频率成分要晚于其他频率成分到达接收端,这一效应称延迟失真或频率失真。

(3) 噪声 在所有传输系统中,不希望有的、插在传输系统和接收端的信号被称为噪声,它是主要的内部损耗。噪声有热噪声、串扰噪声和脉冲噪声三类。热噪声是由导体中电子的热扰动引起的,它存在于所有电子器件和传输介质中。它是温度变化的结果,但不受频率变化的影响。热噪声也称白噪声或高斯噪声。串扰噪声是一种信号路径之间不希望有的电器耦合现象。特殊的电缆屏蔽可以解决某些串扰问题,使用双绞线的一个主要原因就是要消除这种损耗现象。脉冲噪声是非连续的,由持续时间短和幅度大的不规则脉冲或噪声尖峰组成。产生脉冲噪声的原因多种多样,其中包括电磁干扰以及通信系统的故障和缺陷,在通信系统的电气开关或继电器改变状态时也可能产生。它会干扰话音(言语)传输,但通常不会损失信息。

信噪比

信噪比是信号功率与噪声功率在传输线相同点上的比率。它用来表示信号电平超过噪声电平的量值,以分贝计。

分贝表示功率输出与功率输入之比。就信噪比来说,分贝用来表示信息信号相对于信号噪声的相对强度,用 S/N 表示, S 为信号强度, N 为噪声强度。

高信噪比意味着噪声电平大大低于信号电平,这样,噪声不会干扰所发送的信息。反之,低信噪比意味着噪声电平高于信号电平,会产生严重的数据失真。

在话音通信中,30 dB 的信噪比是令人满意的。信噪比越高,信号在接收端就越清晰。

参考文献

1. 胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002
2. Sheldon T. LAN Times Encyclopedia of Networking. McGraw-Hill Inc., 1994 (胡道元)

chuan chuli yuyan

串处理语言(string processing language)

一类用于描述串处理的语言,即一类字符处理的语言。在程序设计语言中,“串”这一名词通常指字符序列,例如,ABC 是三个字符组成的串。大多数情况下计算机的输入形式是字符串的形式(例如在终端上键入的命令),同样,计算机的输出大多数也是串的形式。

常规程序设计语言的主要职能集中于数值计算和数据处理,然而,其中总要含有许多重要的串处理工作。例如,编译程序是以字符串作为输入,分析该字符串并生成二进制机器码或字符串输出。命令解释程序也总是分析命令串,并且执行相应的操作。这类程序是经常且广泛使用的程序,显然该类程序必须是高效的。正是基于这一原因,该类程序一般均采用系统程序设计语言来书写,如 C 语言等,而非高级串处理语言。但是,高级串处理语言对诸多复杂问题的处理提供了良好的支持,如:语言转换、计算语言学、计算机代数、正文编辑以及格式化文档生成等等。

1957 年—1958 年由 V. Yngve 设计的 Comit 为第一个串处理语言,随后出现多种串处理语言,例如:SNOBOL, Ambit, Axle, Convert, Panon, Icon 等等。

关于数值计算中的数学表示,至今在串处理中既未对其应具有的操作形成统一的认识,也未对串处理有一个标准的记号表示,串处理语言的开发人员几乎是在无先例的情况下开始工作的。因此,串处理语言的记号,程序结构以及对问题的表示研究均与传统的程序设计语言有本质的不同。

目前,串处理中有 4 种基本操作已广为接受:并置、子串的识别、模式匹配以及串的转换。并置是在一个串之后添加另一个串来构成一个长的串的操作。因此,对串 AB 和 CDE 进行并置的结果是 ABCDE。这一操作是将串看作是字符序列这一概念的自然扩展。一个串称为子串的定义是它完全含在另一个串中。例如:BC 和 CDE 均是 ABCDE 的子串。

最重要的串操作是模式匹配:检查一个串,在其中确定对子串的定位,并确定该串是否具有某种性质。例如:确定一个特定的子串的出现,出现的位置,以及子串间的特定关系等。串的转换同模式匹配息息相关,它通过一个串进行模式匹配的结果,再形成对子串的替换。

最流行的串处理语言为 SNOBOL。

参考文献

Griswold R E, Poage J F, Polonsky I P. The SNOBOL4 Programming Language 1971. Prentice-Hall Inc. (郑国梁)

chuanxing chuanshu

串行传输 (serial transmission) 发送方传到接收方的数据在同一时间内只有一位的传输方式。与同时可传输多位数据的并行传输相比,串行传输的速度要慢,但比较经济实用。

在进行并行数据传输时,需要一根至少有 8 条数据线(一个字节是 8 位)的电缆将两个通信设备连接起来。当进行短距离传输时,使用这种方法的费用还是可以容忍的。但是,在进行长距离数据传输时,其费用要远比串行传输高昂得多,因此,长距离数据传输一般都采用串行方式。

在串行传输的发送端,具有 8 位总线的个人计算机(PC 机)内的发送设备将 8 位数据同时送给并串转换部件,将数据从并行方式转换为串行方式,然后数据以串行方式逐位到达接收端的设备中。在接收端,为了进行处理,PC 机的串并转换部件则需将数据从串行方式转为并行方式。这一过程看起来容易,实际上相当复杂。在串行传输中,硬件和软件必须密切配合才能进行传输。

串行传输按传输的方向可分为三种:

(1) 单工通信 只支持数据在一个方向上流动。

(2) 半双工通信 允许数据双向传送,但在同一时间内,只允许数据在一个方向上传输。半双工通信优于单工通信,因为它可允许数据在两个方向上传输,是可切换方向的单工通信。

(3) 全双工通信 允许数据同时在两个方向上传输,因此全双工通信是两个单工通信方式的结合,它要求发送设备和接收设备都有独立的接收和发送能力。

串行传输的另一个重要方面是数据传输的定时和数据接收的定时。在并行传输中,8 位或更多位的数据同时从源地送往目的地,各个数据位是很容易鉴别的。在串行传输中,各个数据位逐次从源地送到目的地,这就要求在数据源和数据目的地之间进行同步,以便将各个数据位、字符和报文区分开来。

串行传输按采用的同步技术可分为异步串行数据传输和同步串行数据传输(参见异步传输与同步传输)。

在串行数据传送中,数据传送速率是用每秒传送的位数来确定的,如每秒传送 9 600 位,则称其传送速率为 9 600 b/s。

参考文献

胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002 (胡道元)

chuangkou xitong

窗口系统 (window system) 通过窗口来控制计算机显示器及输入设备的一种系统软件。窗口是指显示屏幕上的一块矩形区域,它显示用户或系统某一进程的输出。窗口又可看作是一种虚拟终端,一个屏幕上可有多个窗口,显示多个输出,不同窗口可互相重叠。窗口系统所管理的资源有常用的窗口、图符、选单、指点设备,即 WIMP,还有屏幕、像素映象、色彩表、字体及光标等。窗口系统是图形用户界面的基础,它为用户与计算机对话提供了窗口界面、编程接口及窗口管理接口。20 世纪 80 年代初美国施乐(Xerox)公司 Alto 计算机上运行的 Smalltalk-80 程序设计环境是第一个实用的多窗口系统;苹果(Apple)公司的 Macintosh 微型计算机最早在操作系统上使用窗口界面;微软(Microsoft)公司的 Windows 是 PC 机上最重要的窗口系统;麻省理工学院(MIT)等开发的 X 窗口系统广泛应用于各类计算机,其版本 X. 11 已成为事实上的工业标准。

窗口系统按内部结构可分为基于核心及基于客户-服务器两类。前者把窗口系统核心放在操作系统内,其运行效率高,但可移植性差;后者把窗口系统核心及窗口应用程序均作为操作系统的用户进程予以运行及相互通信,它具有网络透明、易扩充、易移植等特点。图 1 为基于客户-服务器窗口系统的

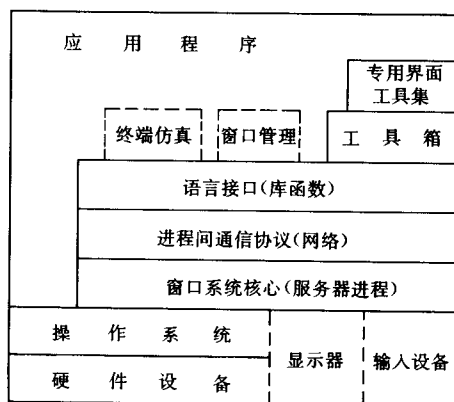


图 1 窗口系统的结构层次图

结构层次图,其中窗口系统核心为服务器进程,用来对显示器及输入设备进行操作;应用程序、终端仿真、窗口管理均作为客户进程通过网络上进程间通信协议与服务器进程通信;语言接口、工具箱及专用界面工具集分别向程序员提供低级、高级及具有某种风格的编程接口。窗口系统的产生极大地影响了操作系统、用户界面的发展,目前正朝着支持三维图形、支持多媒体及使用硬件芯片提高其性能等方面发展。

参考文献

1. 董士海. 窗口系统的内部结构及标准化. 计算机辅助设计及图形学学报, 1990, 2(2): 35 ~ 40
2. Scheifler R W, Gettys J. The X Window System. ACM Transaction on Graphics, 1986, 5(2): 79 ~ 109
(董士海)

cifa fenxi

词法分析 (lexical analysis) 逐个读入源程序字符并按照构词规则切分成一系列单词,再转换成词标流的过程。单词是语言中具有独立意义的最小单位,包括保留字、标识符、运算符、标点符号和常量等。词标是单词的机内表示,其格式由实现系统规定。

词法分析是编译过程中的一个阶段,在语法分析前进行。可以作为单独的一遍,将源程序转换成词标流供下一遍使用。也可以和语法分析结合在一起作为一遍,由语法分析程序调用词法分析程序来获得当前词标供语法分析程序使用。

在词法分析程序的设计和实现中,首先需要描述和刻画语言中的原子单位——单词,其次需要识别单词和执行某些相关的动作。描述程序设计语言的词法的机制是 3 型文法和正则表达式,识别机制是有穷状态自动机。

在词法分析过程中,与语法分析无关的单词应预先处理。如为增加易读性而在保留字中引进的任选字,与语法规义分析无关,处理时可掠过,无需产生相应的词标。

参考文献

- Aho A V, Sethi R, Ullman J D. Compilers Principles, Techniques and Tools. Addison-Wesley, 1986
(钱树人)

ci cunchuqi

磁存储器 (magnetic storage) 以硬磁材料为

存储介质的计算机存储设备。

在外磁场中的硬磁材料会被磁化,当外磁场消失后,硬磁材料仍保有磁性,其磁化方向与外磁场方向相同。因此可以用来作存储介质。

硬磁材料保有的磁性只有两个方向,非常适合用来存储二进制的数字。做成的存储器有两种结构,一是固定式,存储介质与外加磁场间没有相对运动。如**磁心存储器**、**磁杆存储器**和**磁膜存储器**。另一是运动式,记录和读出时存储介质与外加磁场间有相对运动。如**磁鼓存储器**、**磁带存储器**和**磁盘存储器**。

磁心存储器 用磁心作为存储元件的存储器。磁心是用铁氧体材料经过压制、烧结而成的环状磁性物体。磁心的尺寸大约是 $0.8\text{ mm} \times 0.6\text{ mm} \times 0.2\text{ mm}$ (外径 \times 内径 \times 高)。每一个磁心可存储一位二进制数,根据存储器容量要求,把许多个磁心组成一个矩阵,例如 32×32 的矩阵,一个磁心相当于一个字中的一位。再把许多块板叠成一个立方体,板数等于字长的位数加上检验的位数,这样的立方体称为磁心体。它的存储量就是 1 024 个字。磁心体和外围电路(译码电路、驱动电路、读出放大电路和控制电路等)组成一个磁心存储器。

第一个采用铁氧体材料的磁心存储器在 1953 年由美国麻省理工学院研制成功。由于它在性能、成本、体积各方面都优于当时的其他存储器,很快成为内存储器的主流产品。磁心本身可自动化生产。但穿线仍要人工进行,成本高。磁心本身不可能做得非常小,存取速度也慢,体积也大。因此,在半导体存储器研制成功后,磁心存储器很快就被淘汰了。

磁杆存储器 利用杆状磁性材料作为存储介质的半固定或固定存储器。其结构是在平板上布有许多绕在一起的初级绕组和次级绕组,每个绕组存储一位。磁杆用软磁材料制成。当绕组中间插有磁杆时,初级绕组和次级绕组间有耦合,电感大,表示这一位是 1;绕组中没有磁杆,表示这一位是 0。

磁杆存储器是一种只读存储器。随着半导体只读存储器(ROM)的研制成功并大量生产,磁杆存储器因在结构、体积、性能、成本等方面都不能与之相比而被淘汰。

磁膜存储器 利用薄膜磁性材料作存储介质但是没有运动的存储器。磁膜存储器采用类似于磁心存储器那样的电流重合法进行选址、写入和读出,作为存储介质的材料是铁镍合金,用电镀、蒸发、溅射、沉积等工艺在玻璃或其他材料上制成薄膜,膜厚约

80~90 nm。面积约 1 mm^2 。由于磁膜是平面结构,可以用印制电路工艺来制作位线、字线和读出线,曾被认为可以代替磁心存储器。但实际上由于磁膜信号小,干扰大,器件体积大,工艺困难等原因,并没有得到广泛应用。

磁存储器中得到广泛应用的是记录和读出时存储介质与外加磁场间有相对运动的运动式磁存储器。在19世纪中叶,钢丝录音机就已发明。后来用磁带代替钢丝作存储材料的磁带录音机出现,并得到广泛应用。20世纪40年代至50年代研制的早期的计算机也曾采用磁带录音机作存储设备。

运动式磁存储器的存储介质是在带状、圆柱状或圆片状的基体的表面上附着的薄薄的一层磁性材料。因此也称为**磁表面存储器**。它仍是以磁性材料的两种不同剩磁状态或不同剩磁方向来表示二进制数字信息。

磁表面存储器的信息记录和读出过程都是电磁转换的过程,是通过磁头和运动着的磁介质来实现的。

磁头用软磁材料做铁心,上面绕有读写线圈。磁头铁心上有一条很窄的前间隙。工作时,磁头的前间隙靠近磁介质表面,附着有磁介质的基体相对于磁头做匀速运动。写入过程是将数字代码以脉冲电流的形式输入到磁头线圈中,形成磁场磁化磁介质,并以磁化状态的形式保存在磁介质上。读出过程则是通过磁头将磁介质上的磁化状态转换成电信号,再还原成数字形式。

磁表面存储器有**磁鼓存储器**、**磁带存储器**、**硬磁盘存储器**和**软磁盘存储器**等。

磁鼓存储器 利用高速旋转的圆柱体的磁性表面作存储介质的存储设备。磁鼓存储器在20世纪五六十年代是计算机的主要外存储器。随着磁盘存储器的出现与发展,磁鼓存储器就被淘汰了。

磁带存储器 利用稳速运动的带状物体的磁性表面作存储介质的存储设备。磁带性能稳定,能长期使用和保存,而且价格低廉。磁带可以更换,可以脱机存储,因此适用于数据交换、文件档案的复制和保存,以及数据的后备存储,还可用于数据输入。

硬磁盘存储器 利用高速旋转的盘状物体的磁性表面作存储介质的存储设备。从1956年第一台硬磁盘存储器IBM 350问世以来,由于它的存储容量大、数据传输速率高,一直是外存储器的核心设备。随着技术的不断进步,它的存储容量、存取速度和可靠性都有显著的提高,品种也非常丰富。

软磁盘存储器 利用高速旋转的薄片状物体的磁性表面作记录介质的存储设备。1972年研制成功单面单密度的200 mm(8 in)软磁盘机。随后又制成了双面倍密度的130 mm(5.25 in)软磁盘机。1981年日本研制成90 mm(3.5 in)软磁盘机。由于软磁盘的存储容量有限,软磁盘存储器的使用范围受到限制。

磁表面存储器的技术进步是和**数字磁记录**的研究分不开的。提高水平磁记录密度的途径有:采用高矫顽力和高剩磁化强度的磁介质,减薄磁层,减小磁头浮动高度,减小磁头缝长等。1976年日本的岩崎俊一提出了垂直磁记录理论,其记录密度比传统的水平磁记录密度高10倍以上。

磁表面存储器之所以能长期存在,是因为它具有许多优点:存储的内容不易丢失,可长期保存;存储密度不断提高,存储容量大,能满足计算机的发展需要;写入、读出的速度快,而且可以覆盖写。磁表面存储器也有其缺点:有机械运动部件,易磨损,可靠性差,寿命有限;功耗大;体积大。随着新型磁性材料的出现,磁头材料和结构的改进,新的记录方法的采用,在今后若干年内,磁记录的存储密度仍会有成倍的增长,磁表面存储器作为外存储器的主力地位,仍然是不可替代的。

近年来,**自动盒带库**、**磁盘阵列**这一类集成式磁存储器也在迅速发展,磁盘阵列已得到广泛应用。

(林兼)

cidai cunchuqi

磁带存储器 (magnetic tape storage) 利用数字磁记录原理,通过电磁变换传感器件(通常称为磁头)在稳速传动的带状磁性媒体表面进行数据记录的顺序存取存储设备。它通常由磁带传动部件和磁带控制器两部分组成。前者习惯上也称**磁带机**,或称**磁带驱动器**。一般称开盘磁带设备为**开盘式磁带驱动器**,盒带设备为**盒带驱动器**。

磁带传动部件有两项基本功能:一是使磁带按一定速度平稳地通过磁头的读写缝隙;二是对磁带进行写入或读出,即从控制器送来的数据电信号经磁头工作缝隙对磁带进行磁化(写入),或由磁头将从磁带感应到的磁束变化转换成电的信号送往控制器。其基本构成部分为:①读写磁头组件及读写电路;②使磁带快速启停并稳速行走的主动轮、驱动电机及伺服电路;③带盘电机及伺服电路;④磁带的缓冲机构;⑤防止磁带行走时左右晃动的导轮或

导柱等导向机构。磁带控制器是按照主机发来的命令控制磁带机进行数据的写入或读出的电子设备。一台磁带控制器可连接多台磁带机,构成磁带存储子系统,但在进行读写操作时只能选择其中的一台进行读写。

磁带存储器作为计算机的一种辅助存储器已有悠久的历史,在 20 世纪 40 年代后期最早问世的几台计算机中就已使用磁带存储设备,其技术源于当时的录音设备,磁带从供带盘到收带盘直接传动,设备简陋。1953 年 IBM 公司推出了利用真空缓冲带箱的 726 型磁带机,将大转动惯量的带盘驱动从保持磁带快速启停与稳速传动的主动轮隔离开来。从此,磁带驱动设备开始了新的局面,并奠定了一直沿用至今的半英寸带宽的通用标准。经过 20 年几代技术的改进,位密度从 100 bpi 增至 6 250 bpi (246 b/mm),带速从 75 in/s (1.9 m/s) 增至 200 in/s (5.08 m/s),数据传输速率从 7.5 kb/s 增至 1 250 kb/s,记录块间隔从 19.05 mm 减小到 7.62 mm。

在继续提高记录密度的同时,想要进一步减小记录块间隔与进一步提高带速已都很困难,因为主动轮和带盘的大转矩、低惯量伺服电机的设计已几乎到达极限。1984 年 IBM 3480 盒带驱动器问世后,记录块间快速启停的开盘磁带机逐步被新型流式连续传动的盒带驱动器取代。随着微型计算机的发展,小型盒带存储设备有更迅速的发展。常见的小型盒带驱动器主要有采用纵向数字磁记录技术的 1/4 英寸 QIC 盒带驱动器与采用螺旋扫描磁记录技术的 4 mm 数字音频盒带 (DAT) 驱动器和 8 mm 视频盒带驱动器。小型盒带驱动器属低速型设备,通常均为单个磁道串行工作方式,设备小巧,适用于微型计算机。其单盒容量和记录密度高。经加长磁带长度、采用数据压缩技术、提高磁道密度以及新的信号处理技术后,这类小型盒带的容量又有明显提高。(参见盒带驱动器)。

磁带驱动设备有许多类型,从不同角度分,大体上可分为大型计算机用的高性能快速型与微型计算机用的低速型,开盘带型与盒带型,快速启停型与流式型等。

磁带驱动器广泛用于各类计算机中,起着如下作用:①保存不经常存取的(例如每周仅 1~2 次,或甚至不到 1 次)带有永久性的文件资料;②作为磁盘等直接存取存储设备的备份;③计算机间的数据交换;④键输入数据的初始收集,并继而作为计算机的输入媒体。

参考文献

1. Harris J P, Phillips W B, Wells J F et al. Innovations in the Design of Magnetic Tape Subsystems. IBM J. Research and Development, 1981, 25 (5): 691 ~ 699
2. Phillips W B. Trends in High-Density, Digital, Magnetic-Tape Recording. Sixth IEEE Symposium on Mass Storage Systems, 1984: 72 ~ 75 (刘锡刚)

ciguangpan qudongqai

磁光盘驱动器 (magneto-optical disc drive)

一种利用磁光效应的可改写光盘驱动器,是继磁带机、磁盘机之后,在 20 世纪 80 年代中后期发展起来的一种新型的计算机外存储器,也叫磁光盘机。它具有存储容量大,存储媒体可换,以及可顺序存取也可随机存取等特点。主要用于信息的海量存储,常作为小型计算机或高档微型计算机的配套外设,尤其适用于多媒体计算机系统中。由于磁光盘机的写、读是通过磁和光的共同作用来完成的,因此在名称上有“磁光”字样。由于记录媒体采用“居里点”高于常温的硬磁材料,在常温下其磁化状态不能被外界偏置磁场所改变。但是,当聚焦激光的能量使记录媒体表面某个极小的局部区域温度上升到“居里点”以上时,这个小区域内的媒体就会随外界偏置磁场的变化而迅速被磁化并达到饱和,这样就可像磁盘一样进行写入或抹除信息。信息的读出是利用线性偏振光的克尔 (Kerr) 效应来实现的。当线性偏振光入射到记录媒体上并反射时,会根据媒体上记录的信息 1 或 0 所形成的不同磁化状态,使反射光的偏振发生不同方向的旋转,这种现象就是所谓的克尔效应。两种不同磁化状态所对应的反射偏振光的偏振方向之间的夹角称之为克尔角。在磁光盘机中,信息的读出实际上就是检测克尔角。磁光盘机为了按上述基本原理实现信息的写入和读出,它必须包括下述主要结构:带动光盘旋转的主轴电机,完成用激光向光盘上写入和从光盘上读出信息的光盘头,配合信息的抹除和写入用来产生不同方向偏置磁场的写抹电磁铁,带动光盘头寻找记录道的直线电机,把光盘头定位在记录道上并保证激光束聚焦在光盘表面的二维执行机构,以及控制磁光盘机工作的电子线路板等。光盘直径尺寸目前主要有 130 mm (5.25 英寸) 和 90 mm (3.5 英寸) 两种。随着技术的不断发展,还会出现更小尺寸的磁光盘机。衡量磁光盘机性能和影响其使用的主要技术指

标有:容量、数据传输速率、平均存取时间、旋转速度、接口、外形尺寸和平均无故障时间等。以目前市场上常见的日本理光公司的 RO-5031E 型 130 mm 磁光盘机和日本富士通公司的 M2511A 型 90 mm 磁

光盘机为例,主要技术指标如表 1 所示。磁光盘机的发展趋势是增大容量,减小尺寸,提高数据传输速率,加快存取时间,解决盖写功能等。

表 1 磁光盘机目前主要技术指标(1993 年)

	容量 /MB	数据传输速率 /Mb/s	平均存取时间 /ms	旋转速度 /r/min	接 口	尺 寸	平均故障 间隔时间 /h
90 mm 盘径	128(单面)	8.8	43	3 600	SCSI-2	101.6 mm × 146.0 mm × 25.4 mm	30 000
130 mm 盘径	650(双面)	11	37	3 600	SCSI	146.1 mm × 203.2 mm × 82.6 mm	30 000

参考文献

杨建东,裴先登.可重写光盘机与多功能光盘机技术.电子计算机外部设备,1993(3):38~42

(高宝石)

cikaji

磁卡机 (magnetic card reader) 利用磁记录原理从磁卡上读出或写入数据的设备。

磁卡是一种在基片上带有条状磁存储媒体的卡片。常用的基片材料有经过化学处理的纸、聚氯乙烯和聚酯。条状磁性媒体称为磁条,用来存储信息,一般宽 5~10 mm。制作磁条可以将磁性材料直接涂布在基片上,也可以将磁带粘贴在基片上。在磁条表面涂有保护膜,因此有的磁卡看不见磁条。磁卡的其他部分根据用途和需要印有文字或图形。磁卡的大小为 85.5 mm × 54 mm,厚度有两种:薄的为 0.30 mm,厚的为 0.76 mm。

磁条的记录原理和磁带、磁盘相类似。它的磁性能指标:矫顽力 $H_c = 300 \sim 800 \text{ Oe}$ ($1 \text{ Oe} = 79.5775 \text{ A/m}$),剩磁感应强度 $B_r = 1000 \text{ Gs}$ ($1 \text{ Gs} = 10^{-4} \text{ T}$)。ISO 标准规定了磁条上三条磁道的记录密度和容量,见表 1。

表 1 磁条的记录密度和容量

磁道号	记录密度/bpi	记录字符容量/个
1	210	78
2	75	38
3	210	106

注: bpi 即位每英寸。

根据不同的需要,磁卡可以采用 1 条、2 条或 3

条磁道。磁条上记录方式通常采用双频制(FM 制)。FM 制有自同步,读出数据不致因速度变化而影响正确性。数据是以串行的方式记录在磁条上的,所记录的格式和内容则根据需要由发卡单位自行决定。不同应用场合的磁卡,记录的格式互不相同,以避免混淆。

磁卡有两种。一种是只读的,磁条上的信息内容是预先写入的,此内容只供辨别持卡人身份用,和条码相类似。信用卡是一种常见的只读磁卡,它拥有信息最少的情况是只有持卡人编号和发卡银行的标志。另一种是读写式,既可读出也可写入,如地铁车票磁卡、电话磁卡。磁卡上写的款数在每次使用后都要修改。

磁卡机由四个主要部分组成:卡片传送机构、位置检测机构、磁头和电子电路。卡片传送机构有手动、电动两种。其功能是带动磁卡以 $100 \sim 1000 \text{ m/s}$ 的速度向前运动使磁条与磁头接触,读出或写入信息。对于手动式,利用人手拉动把磁卡沿指定的轨道移动,机构简单,但运动速度变化大。另一种是电动式,用手把磁卡插入磁卡机插口时,位置检测机构检测到有卡,就启动电机带动磁卡向前运动,读写完毕后,电动机反转,把磁卡沿原路退出。它的机构复杂,但运动速度稳定。只读式磁卡机只有读磁头。读写式磁卡机则有读头,也有写头。电子电路除了读出电路、写入电路外,还有与计算机接口的电路。磁卡机通常通过 RS-232C 接口与计算机连接。

和条码阅读器一样,磁卡机是一种专用的输入设备。读写一张磁卡只要一秒左右的时间,比用键盘输入快并且可靠。它使用方便,而且其数据保密性好,可改写性也好,因此,得到广泛应用。如金融、

保安、商业、交通等方面,都有各种各样的磁卡在使用。常见的有信用卡、地铁车票卡、电话磁卡、考勤卡、电子锁、通行证以及磁卡电表等。特别是作为一种电子货币,在金融、商业上发挥着越来越大的作用。

由于磁卡的磁性易被破坏,在可靠性方面不如 IC 卡,在存储容量方面也不如 IC 卡,在许多应用领域已被 IC 卡所取代。(陆源茂)

cipan cunchuqi

磁盘存储器 (magnetic disk storage) 在恒速旋转的圆形磁性媒体表面沿同心环形轨迹,通过磁头电磁转换器件进行数据记录的直接存取存储设备。又称为旋转磁表面型存储设备。磁盘存储器由磁盘驱动器和磁盘控制器两部分组成。磁盘驱动器的功能是驱动盘片按一定转速稳速旋转,驱动载有磁头的头臂到达且稳定在指定的半径位置上,控制磁头在盘面磁层上按一定的记录格式和编码方式进行写入和读出。固定头磁盘驱动器则无动臂动作。不论是可换盘硬磁盘驱动器、固定硬磁盘驱动器还是软磁盘驱动器,作为驱动器的工作原理和设备结构是大同小异的。(参见**硬磁盘驱动器**、**软磁盘驱动器**)。

磁盘控制器是控制磁盘驱动器实施数据的存入与取出的设备。它按照主机发来的命令控制驱动器进行磁道的寻找(头臂定位)、头的选接、数据的写入(存)与读出(取)等操作(参见**硬磁盘控制器**与**软磁盘控制器**)。

与磁盘存储器有关的参数和性能指标如下。

存储面数 指可存储数据的盘片的表面数。对于软磁盘驱动器,存储面数为 2。对于硬磁盘驱动器,存储面数是盘片数的 2 倍。但有的硬磁盘驱动器采用伺服面技术,占了一个盘面。例如,5 片的硬磁盘,存储面数只有 9 面。一般每面只用一个磁头,存储面号也就是磁头号。

磁道 磁盘存储表面被分成许多同心圆,每一个同心圆就是一个磁道。磁道有一定宽度,是存储信息的区域。

扇区 每个磁道所在的同心圆分成若干等份的段,每一段称为一个扇区,每个扇区存储的字节数是一定的。软磁盘驱动器的各个磁道的扇区数是相等的。硬磁盘驱动器在等位密度记录的情况下,一个盘面各个磁道的扇区数是不相等的。盘面内侧直径小的磁道,扇区数少;盘面外侧直径大的磁道,扇区

数多。

道柱面 在硬磁盘驱动器中,把所有的盘面上的同一直径的磁道看成一个整体,就像一个圆柱体的外表面一样,称为道柱面,简称柱面。

存储密度 存储密度常用道密度、位密度和面密度来评定。

(1) 道密度 道密度是指沿磁盘径向方向,在单位长度内的磁道数目。道密度习惯上用每英寸的磁道数来表示,记作 tpi;或用每毫米的磁道数来表示。

(2) 位密度 位密度是指沿磁道圆周方向单位长度内所存储的二进制数的个数。习惯上用每英寸存储多少位来表示,记作 bpi;或用每毫米存储多少位来表示。

(3) 面密度 面密度是指单位面积上存储的二进制数的个数,它等于道密度与位密度两者之乘积。

存储容量 存储容量是指磁存储器所能存储的二进制数的总量,以位数或字节数计量。常用单位为 KB、MB 和 GB。

存储容量有未格式化和格式化两种。未格式化容量指的是,在磁盘机的全部存储面上所有磁道所能存储的二进制码的总量。所谓格式化,就是指计算机系统按要求在数据存入之前在磁道上先写入地址、识别码和同步码等与地址有关的信息,目的在于能迅速辨认所存取的数据所在位置。格式化容量按下式计算:

格式化容量 = (存储字节数每扇区) × 扇区总数

对于各个磁道的扇区数是相等的磁盘,其格式化容量可按下式计算:

格式化容量 = (存储字节数每扇区) × (扇区数每磁道) × (磁道数每面) × 面数

显然,格式化后,与地址有关的信息要占去一定的存储容量,格式化容量低于未格式化容量。

存取时间 存取时间是指主机发出指令到数据传输结束所需的时间。即磁头从当前所在位置移动到目标位置(目标磁道)并稳定下来,然后从目标磁道上寻找要读写的扇区并进行读写所需的全部时间。它由寻道时间、等待时间和传输时间三部分组成。

寻道时间 是磁头运动所需的时间。显然,运动距离最小即寻找相邻磁道的寻道时间最短,而运动距离最大即从首道寻找到末道(或相反)的寻道时间最长。前者称为最小寻道时间,后者称为最大寻道时间。平均寻道时间是采用统计平均值求得的,

一般等于运动距离为最大距离的 $1/3$ 时所需的时间。

等待时间是指磁头到达目标磁道后等待要读写的扇区到达磁头下方所需的时间。最短是0,最大是磁盘旋转一周所需的时间。显然,平均等待时间是磁盘旋转一周所需时间的一半。

传输时间是指数据从内存写到盘面或从盘面读出送到内存所需的时间。传输时间在存取时间中所占比例很小。

数据传输速率 **数据传输速率**是指在单位时间内磁存储器传送的二进制数的位数或字节数。数据传输速率的单位为 Mb/s 或 MB/s。

硬磁盘存储器的数据传输速率有外部数据传输速率和内部数据传输速率之分。外部数据传输速率指硬磁盘驱动器与主机之间的数据传输速度。它与所采用的接口有关。硬磁盘存储器的内部数据传输速率是盘面与高速缓存之间的数据传输速度。它等于位密度与磁道线速度的乘积。

磁盘按基片材料分为硬磁盘与软磁盘两类。硬磁盘驱动器分固定磁头(或每道一头)与可动磁头两类,后者又可分为可换盘硬磁盘驱动器与固定硬磁盘驱动器两类。可动磁头固定硬磁盘驱动器是广泛使用的类型。软磁盘驱动器则只有可动磁头、可换盘一种类型。

磁盘驱动器的存储密度和存储容量还将会有大幅度的增长。磁盘存储器将继续在计算机辅助存储设备中占重要地位。

参考文献

Wood R W. Magnetic Megabits. IEEE Spectrum, May, 1990: 32 ~ 33, 36 ~ 38 (刘锡刚)

cipan zhenlie

磁盘阵列 (magnetic disk array) 用多台磁盘存储器按数据分块与冗余信息容错,以矩阵形式组成的快速大容量外存储子系统。它在阵列控制器的组织管理下,能实现数据的并行、交叉存取存储操作。由于阵列中的一部分容量存放有冗余信息,一旦系统中某一磁盘失效或存取通道失效,利用冗余信息可以重建用户数据。磁盘阵列是一个包括许多品种的泛称,常见的一种称为廉价冗余磁盘阵列(RAID)。构造磁盘阵列的思路适用于构造其他存储器阵列,如固态盘阵列、只读光盘(CD-ROM)阵列和磁带存储器阵列。

磁盘阵列是在中央处理器性能逐年增强,而输

入输出速度受限,存储容量又与日俱增的背景下产生的。磁盘阵列的性能在许多方面超过单台大型存储设备的性能,而价格则低于同容量的单台大型设备。因此得到很大发展,除应用于大型计算机外,还广泛应用于工作站等。

磁盘阵列的构思渊源于早期提出的拆分、交叉存取、分块等概念。拆分的意思是将数据分割为小条,按条并行存取;交叉存取是指在多台磁盘存储器上进行交叉、并行操作;而分块的含义则是对群集的数据分成较大的块,将大块数据分布到若干台磁盘存储器上。它们的含义有相似之处,但存在着某种差别。拆分是针对主机请求读、写数据而言的,交叉存取是从减少存取时间的意义上说的,而分块则是针对存储空间的有效利用而采取的。三者分别用于处理三个层面上的并行性问题。

1987年,美国 D. Petterson 等人把廉价冗余磁盘阵列(RAID)分成5级,即 RAID 1 ~ RAID 5。后来为方便比较分类,人们增加了 RAID 0,共为6级。后来有各种新的 RAID,如能检验双盘错并恢复丢失数据的 P+Q, RS, EVENODD 等构造出现,但未形成公认的级别。

0级冗余磁盘阵列(RAID 0) 一种不具备容错能力的阵列。阵列的平均无故障时间(MTTF)只是单台磁盘存储器的 N 分之一(N 为构成阵列的磁盘存储器的总数)。显见,0级盘阵列的可靠性最差,但容量与数据传输速率则为单台磁盘存储器的 N 倍。此种盘阵列可用于扩大存储容量和提高输入输出速率。

1级冗余磁盘阵列(RAID 1) 采用镜像容错改善可靠性的一种磁盘阵列。阵列中磁盘分成若干组,每一组由两个相同的盘组成。每次写数据时同样地写在一组中的两个盘上,因此其可靠性很高。它的 MTTF 远大于单台磁盘存储器,而且在某一磁盘存储器失效后,经更换新的之后,数据可以重构。但是磁盘阵列的有效容量减少到只有总容量的一半。

RAID 1 虽然结构简单,但可靠性高,随着计算机硬件价格的降低,很多硬件容错计算机都采用了这种镜像磁盘结构。RAID 1 常用于出错率要求极低或不允许出错的应用场合,例如某些事务处理(财政、金融等)系统中。

2级冗余磁盘阵列(RAID 2) 采用汉明码作错误检验的一种磁盘阵列。汉明码是一种 (n, k) 线性分组码, n 为码字的长度, k 为数据的位数, r 为用于检验的位数,即因监督错误使用的冗余位数, $r = n - k$, $2^{r-1} \geq k + r + 1$ 。

在构造此种阵列时,采取在每组 n 台磁盘存储器中设置 r 台用于检验。例如,若每组有 14 台磁盘存储器,则设置 4 台用于检验;若每组为 25 台,则设置 5 台用于检验,如此类推。这种分配符合汉明码的编码规则。RAID 2 与 RAID 1 相比减少了检验用磁盘存储器的数目,因而扩大了有效容量。再则,由于不再需要附于每个扇区之后的循环冗余检验码,降低了约 10% 的容量损失。

按位交叉存取最适于采用汉明码检验。因此,在 RAID 2 中,数据以位或字节为单位分割处理,然后分别存储到各个磁盘相同位置的扇区里,控制器同时将形成的汉明码写入检验盘。对于大批量数据的写入和读出,各组中的每台磁盘存储器同时操作,因而加快了存取速度。但若写入只是少量的部分数据,则需分三步进行:首先读出所有数据,然后合并新老数据,最后将数据和检验信息分别写入数据盘和检验盘。这样,磁盘阵列的带宽与输入输出速率将因之降低。所以这种磁盘阵列适宜于用作科学与工程计算的超级计算机,而不适宜于用作事务处理的计算机系统。

3 级冗余磁盘阵列(RAID 3) 减少用于检验的磁盘存储器的台数有利于提高磁盘阵列的有效容量。鉴于磁盘存储器本身多数已采用了循环冗余检验码检测错误,只要在阵列中能够检测出错误出现在哪台磁盘存储器上,便能达到纠正错误的目的,所以检验盘的数目可以减少。3 级冗余磁盘阵列就是采用奇偶检验码将检验盘数目减少到只有一个,它可以识别出一台出错的磁盘存储器。

RAID 3 和 RAID 2 一样采用按位或按字节交叉存储数据。由于阵列中每组只用一台检验盘,视图的规模可降低费用 4% ~ 10%,而其性能与 RAID 2 接近。同样地,不需重读便可即时校正所有软错。

此种磁盘阵列,如同 RAID 2 一样更适合于与超级计算机配合使用,适用于大数据量的科学计算和高数据传输速率的图像处理等方面。对于事务处理系统也因小块数据的读写性能改进不大而效果较差。

4 级冗余磁盘阵列(RAID 4) 一种可独立地按扇区对组内各盘读写的阵列,它也只用一个检验盘。

RAID 3 将一次传输的数据(传输单元)按位或字节分割后,分散记录在组内的相同扇区号的各个磁盘存储器上。其优点是整个阵列的带宽可以被充分利用,使大批量或成组的数据传输时间得以减少。但是,每次读、写都要牵涉到整个组,故每次只能完成一次输入输出,且寻道时间、旋转等待时间趋向最

坏情况。RAID 4 则改变为将整个传输单元存储在一个扇区内。两者的主要差别是:RAID 3 按位或字节交叉存取,而 RAID 4 则是按扇区交叉存取。因而对于小量传输数据可以单独地对某个盘操作,无需像 RAID 3 那样完成一次小量数据写入要涉及全组,RAID 4 只牵涉组中的两台磁盘存储器(一台数据盘、一台检验盘),即对这两台盘读旧数据,读旧奇偶值,经计算后再写入新数据和新奇偶值,因而提高了小量数据的输入输出速率。

在 RAID 4 中,奇偶值的计算比较容易。若已知旧数据(D_{old})和旧奇偶值(P_{old}),当新数据(D_{new})写入时,新奇偶值可由下式算得:

$$P_{new} = (D_{old} \text{ XOR } D_{new}) \text{ XOR } P_{old}$$

RAID 4 只在小批量数据传输上提高了 I/O 速率,有利于提高事务处理系统的效率,其他方面与 RAID 3 相同。

5 级冗余磁盘阵列(RAID 5) RAID 4 虽然改善了读出的并行性,尤其是对于小批量(不大于一个扇区的容量)数据的读出,但对于写数据则仍局限在每组同时只能一次操作。因为每次写数据都必须读、写检验盘,所以检验盘成了改善性能的关键。RAID 5 即为解决这一问题而提出的一种磁盘阵列。它不设置固定的专作检验用的磁盘存储器,而按某种规则在组内临时指定检验盘。于是在同一台盘上既记录有数据,也记录有检验信息。这一改动,解决了争用检验盘的问题。例如,某一 RAID 4 中,1 ~ 4 号磁盘存储器用于数据记录,5 号磁盘存储器用于奇偶值记录。若对 2 号盘的 0 扇区和 3 号盘的 1 扇区写数据,则因都要在 5 号盘上写奇偶值,只能顺序进行。RAID 5 则允许在同一组内并发进行多个写操作。对于上述情形,虽然仍需两次写奇偶值,但可分散在两个盘(4 号和 5 号)上进行,缓解了集中在专用检验盘上写检验字的矛盾。因此,RAID 5 在小量数据写入和小量数据的“读—修改—写”两种操作上接近 RAID 1,而对大批量数据的读、写则与 RAID 3, RAID 4 相同,且有效容量大,开销少。可见 RAID 5 既适用于超级计算机作科学与工程计算,又适用于作事务处理,是一种在速度、容错能力和有效容量方面较好的磁盘阵列。

阵列控制器 一般由微处理器、高速缓冲存储器、接口、适配器等组成。阵列控制器的主要功能是:①完成命令管理,包括接收主机命令,解释和分解命令,形成控制信号,回收磁盘存储器的工作状态信号以及协调盘阵列各部分的工作等;②进行地址

变换,包括数据分割(拆分和分块)和聚合,地址变换,坏块管理;③实现数据的检错、纠错和重构以及最优化调度。为了扩展阵列的功能,还可以在盘阵列控制器中设置某些功能部件,如数据的过滤与排序、数据的压缩与解压等。

此外,不使用单独的控制单元,仅以软件实现上述功能的磁盘阵列称为**软磁盘阵列**。

盘阵列是并行处理技术在外存储器系统中的卓越应用,它巧妙地处理了速度、容量和成本之间的关系。主要问题是,小量数据写入时的效率,与计算机系统连接的并行性,大量(例如1000台)磁盘(或光盘)存储器的阵列构造等。

参考文献

1. 张江陵,冯丹.海量信息存储.北京:科学出版社,2003

2. Patterson D A et al. Introduction to Redundant Arrays of Inexpensive Disk (RAID). Proc. IEEE, 1989
(张江陵)

cunchu baohu

存储保护 (memory protection) 在多道程序和多处理机系统中,为使多个用户在共享主存储器时,能防止某一个用户程序出错而破坏在主存储器中其他用户程序或系统程序而设置的一种功能。

存储保护功能是靠计算机硬件和软件配合实现的。

存储保护功能可有:对计算机工作状态的保护、对存储区域的保护和访问方式的保护。

工作状态保护

工作状态保护是为防止某一个程序出错而影响整个计算机系统工作而设置的一种保护功能。大多数计算机在执行程序时把工作状态分为两种:一种是执行操作系统或系统管理程序时所处的状态,称为**特权状态或管态**;另一种是执行用户程序时所处的状态,称为**非特权状态或目态**。在计算机中规定:能改变机器运行状态和用户对机器中资源使用权限的特权指令,只有在操作系统和系统管理程序中才能使用,也就是说,只有在特权状态下才能执行特权指令。如果用户程序误用特权指令,就可能对整个计算机系统产生严重破坏,因此在发生非特权状态下使用特权指令时,计算机能进行检查,发出错误中断,进行保护。

存储区域保护

存储区域保护是为防止某一个程序对同时存

放在主存储器中的其他程序进行错误访问而设置的一种保护功能。存储区域保护主要有以下几种方法。

(1) **界限寄存器保护** **界限寄存器保护**在分区管理的主存储器中,对每个存储区域设置1个上界寄存器和1个下界寄存器,分别放置分配给1道程序的存储区域的上界和下界地址。在执行程序时,每当进行逻辑地址到物理地址的变换时,均需把变换所得的物理地址同这对上界寄存器和下界寄存器的内容进行比较,如果小于下界地址或大于上界地址,则说明这道程序越出自己的存储区域,这时计算机就要发出越界中断,进行保护。此外,也可以为每个存储区域设置1个基址寄存器和1个限长寄存器,用它们来检查是否越界,实现界限保护。

(2) **键保护** **键保护**通常用于页式或段页式存储器中。对主存储器的每个页面都设有几位叫做**存储键**的编码,对每道程序也各分配了位数与存储键相同的编码,叫做**保护键**。当一道程序要访问主存储器的某一个页面时,先要把这道程序的保护键同被访问页面的存储键相比较。只允许程序访问那些存储键与其保护键相同的页面,如果检查出不一致,就会发生错误中断,进行保护。存储键和保护键都是由操作系统设置的。

(3) **段保护** **界限寄存器保护**和**键保护**都是对访问主存储器时的物理地址出错进行的保护,而**段保护**可以在逻辑地址变换成物理地址之前出错时,检查出错误,进行保护。在段式存储器或段页式存储器中,每个用户只能访问分配给自己的段。段式存储器中逻辑地址分成段号和段内地址两部分,如果段号超出段表总的项数,或段内地址超出这个段的段长(段表项中有段长栏),则说明有错误。同样,在段页式存储器中,逻辑地址分成段号、页号和页内地址3部分,如果段号超出段表总的项数,或页号超出这个段内总的页数(段表项中有页表长栏),也说明有错误。这时会发生错误中断,进行保护。

(4) **环保护** 前面3种保护方法只能使某一个程序不去破坏其他程序,**环保护**可以保护正在执行的程序自己,这对于系统程序尤为重要。环保护是把程序按其对整个系统能否正常工作的影响程度分成几层,每层规定了访问权限,越里面的层,级别越高。一个层只能访问本层或外面级别较低的层,不能访问内部较高层次的程序,如图1所示。这样可以防止低层程序出错影响到高层的程序。在图1中的例子中,把主存储器分为4层,从里到外为环0到

环3。把系统程序中最重要核心部分放在环0中,比较不重要的部分依次放在环1和环2中,用户程序则在环3中(用户程序当然也可以按其重要性分设几个环)。在运算器中设有现行环号寄存器,放置正在运算器中执行的段的环号,它由操作系统设置。当某一个程序从1个段转移到别的段去的时候,或是要从别的段中读写数据时,先要进行检查,如果进入的段的环号比现行环号小,就要发出错误中断,进行保护。

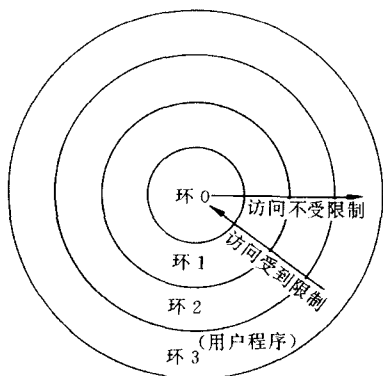


图1 环保护

访问方式保护

对存储区域的保护还可以按不同的访问方式加以保护,即把存储区域进一步区分为是数据区域还是程序区域,是只能读出的区域还是可以写入的区域。上述的几种存储区域保护方法都可以加上访问方式保护,其方法是:对于每个存储区域设置访问方式位(分读、写和执行3种)。读位控制这个存储区域是否允许读出,写位控制它是否允许写入,执行位控制从这个存储区域中读出的数据是否可作为指令来执行。例如,从执行位为0的区域中取出数据当指令执行,就说明有错。同样,如果向写位为0的区域写入数据,亦说明有错。凡检查出这类错误,亦会发出错误中断,以使程序在发生这些错误时不会造成严重后果。

参考文献

孙强南,孙昱东. 计算机系统结构. 北京:科学出版社,1992

(孙强南)

cunchu guanli

存储管理 (memory management) 为了有效地使用计算机系统存储资源而进行的管理。

存储管理是由操作系统在有关硬件的配合下进行的。存储管理包括:存储单元的分配和回收、程序和数在在主存储器与辅助存储器之间的交换、程序再定位、地址转换、存储保护等。

存储管理是随着多道程序设计和分时处理的出现而发展起来的。在计算机系统中,由于中央处理机工作速度快,而输入输出设备的工作速度要慢得多,这一矛盾促成了多道程序和分时处理的出现和发展。要求在同一个处理机上能同时运行多个用户程序,这就需要把存储器划分成多个部分来存放多个程序,因而产生了有效地进行存储管理的需要。

为了保证中央处理机能连续不断地工作,要求主存储器中能存放的程序和数据越多越好。但主存储器的容量不可能无限扩大,因而出现了对换技术。对换是将一个(或一部分)程序从主存储器中取出来写到辅助存储器中,同时将另一个(或另一部分)程序从辅助存储器读到主存储器中去的操作。前一操作称为换出,后一操作称为换入。有了对换功能,就能通过操作系统的自动调度,把主存储器和辅助存储器统一成一个更大的存储器空间。

在存储管理中,需要把存储器划分成多个部分以使多个用户能共享主存储器。一种简单的办法是把主存储器划分成多个地址连续的存储区,给每个用户进程分配1个区。划分是以用户进程为单位的,这时对换也以用户进程为单位进行。

采用分区办法进行存储管理时,由于用户进程长短不等,随着不断进行换入换出,主存储器中会产生很多残片,即夹在两个用户进程之间的零星的空闲存储单元。残片一般不大,放不下1个用户进程,无法加以利用。为了去掉这些残片,就需要时时采取压紧操作,即通过操作系统移动主存储器中的有关进程,把夹在进程间的残片集中在一起,从而可以加以利用。

交换和压紧操作都要改变进程在主存储器中的位置。当进程中的指令在存储器中的位置变动后,指令中的地址码(操作数地址和转移地址)也必须随之变动,这就产生了指令在主存储器中浮动的需要,从而有了把地址分为逻辑地址和物理地址的需要。物理地址又称为实际地址,它是存储单元在主存储器中的实际位置。逻辑地址则是指令或数据在程序中相对于程序开始点的相对位置。逻辑地址是在程序设计中使用的,它只有按一定规则经过变换后才能成为物理地址,只有用物理地址才能找到所需要的存储单元。把逻辑地址按一定

规则转换成物理地址的操作,称为**程序再定位**。如果把指令中的地址码用逻辑地址而不是直接用物理地址表示,就能较好地解决程序在主存储器中浮动的需要。

根据操作时间的不同,把程序再定位操作分为**静态程序再定位**和**动态程序再定位**两种。前者是在程序初始装入主存储器时,由定位装入程序进行的程序再定位;后者则是在程序执行过程中,在对指令或数据进行访问时,才由地址变换机构执行的程序再定位。静态程序再定位靠软件进行,动态程序再定位靠硬件进行。

在分区和交换技术基础上发展起来的存储管理有3种基本方式,即**分页存储管理**、**分段存储管理**和**段页存储管理**。它们采用的存储系统分别为**页式存储系统**、**段式存储系统**和**段页式存储系统**。这些存储系统中的逻辑地址和物理地址之间的转换可参见**虚拟存储器**。分页存储管理和段页存储管理已普遍应用于现代计算机中。

参考文献

孙强南,孙昱东. 计算机系统结构. 北京:科学出版社,1992

(孙强南)

cunchu guanli chengxu

存储管理程序 (memory manager) 操作系统中用于管理计算机**主存储器**及**辅助存储器**的程序。其主要功能有:地址转换、存储分配、存储保护和主存扩充。

(1) **地址转换** 由逻辑地址转换成物理地址。逻辑地址指用户编程序使用的地址,用户程序装入系统后,用户程序涉及的实际存储地址是物理地址。

(2) **存储分配** 各种分配数据结构和分配算法。存储分配与主存区域的划分方式有关。一种是主存被划分成大小不等的连续区域,在这种方式下,存储管理采用的主要技术有:分区存储管理和分段存储管理,因而可采用分区分配或分段分配。分区方式使一个区域可以存放一个作业的连续的地址空间;分段方式使一个区域可以存放一个作业的逻辑分段的地址空间。另一种是主存被划分成大小相等的块,这时可采用**页式存储管理**,使用**页式分配**。它将一个作业的地址空间划分成一连串的面,然后放置到主存的存储块中。

(3) **存储保护** 保护各类程序区和数据区中的信息不被破坏和误用的方法。常用的存储保护方法有:基址限长保护、上下界保护、存储键保护和存储

环境保护。

(4) **主存扩充 虚拟存储及其实现技术**。在具有虚拟存储器的计算机系统中,把物理上独立编址的**二级存储器**——**主存储器**和**辅存储器**连接起来,统一使用。用户作业采用部分装入、部分对换方式运行,即当前使用的程序调入主存,其余仍驻留在辅存,用到时由系统自动调入,从而向用户提供了一个比真实主存大得多的,逻辑上可统一编址的编程地址空间,称**虚拟存储器**。常用的虚拟存储管理方式有:页式存储管理、段式存储管理和段页式存储管理。下面为页式虚拟存储器的基本概念和实现技术:

① **页面** 用户逻辑地址空间划分成若干等长的部分,每部分称为一个页面。页面从0开始依次编号,页面长一般为512字节至4096字节。

② **页框** 主存被划分成大小相同的存储块,每块称为一个页框。页框从0开始依次编号,页框与页面等长。

③ **逻辑地址表示** 在分页系统中,每个逻辑地址用一对数(p,d)来表示。其中,p是页面号,d是虚拟地址在页号为p的页中的相对位移,即页内地址。

④ **页表** 描述页面虚实地址对应关系及页面和页框使用情况的登记表。页表表中至少包含:页面号、页框号、外存地址和中断位等内容。

⑤ **快表** 为了加快虚实地址转换过程,增加一个小容量联想存储器,称**高速缓存**。其中,存放程序执行过程中最常用的那部分页表称为快表。

⑥ **请求调页** 当所需页面不在主存时,发缺页中断请求,暂停正在执行的程序,由系统分析中断源并从外存调入所需页面,修改页表后,让暂停的程序重新运行。

⑦ **页面替换** 出现缺页且主存中无空闲页框时,淘汰主存中的一个页面并从辅存调入所需页面到相应页框。

⑧ **淘汰算法** 又叫置换算法。它是当主存中没有空闲页框,而又要调入新页面时,决定哪一个页面从内存中移走的策略。常用的算法有:先进先出(FIFO)算法、最近最少使用(LRU)算法、最不常用(LFU)算法。

分布式共享主存(DSM)技术近年来发展很快,利用DSM技术可以在无共享主存的分布式多机系统上运行基于共享主存的并行程序。DSM把并行程序中对抽象的逻辑共享主存的读写,转化为对本

地局部主存中物理数据副本的操作,然后通过一致性机制来维护多个数据副本的一致性。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989

(费翔林)

cunchuqi chacuo jiaoyan

存储器差错校验 (memory error checking and correction)

对存储器中存储的信息的正确性所进行的检测和纠错。差错校验建立在信息冗余的基础之上,即在原有的信息中增加 1 位或多位,使之具有检错或纠错的能力。只有检错能力而无纠错能力的通常称作检验。附加的冗余位称为检验位,原有的信息位和检验位在一起合称检验码。

在存储器中常用的是奇偶检验和海明码检验。在奇偶检验中,检验码中只有 1 个检验位。若检验码中的“1”的个数为奇数,称为奇检验码;若为偶数,则称为偶检验码。例:

信息位	奇检验码	偶检验码
11110000	111110000	011110000
01110000	001110000	101110000

奇偶检验的实现分为两步: ①检验码生成。当信息写入存储器时,生成奇检验码(或偶检验码),写入存储器。②检验。当从存储器读出检验码时,检查它是否仍为奇检验码(或偶检验码),如果是,则存储正确;否则,有错。

奇偶检验码只能检测出 1 位错,不能定位,因而不能纠错。汉明码有多个检验位,是一种可以检测出两位错或纠正 1 位错的检验码。它的基本原理是使每一信息位参与多个不同的奇偶检验,如果安排适当,这多个奇偶检验可以惟一地反映出检验码的出错情况。在汉明码中,检验位的位数 K 和信息位的位数 N 可以是下列关系:

$$2^{K-1} \geq N + K + 1$$

现以 $N=10$ 为例,从上式可计算出 $K=5$ 。令 10 位信息位为 $a_i (i=1, 2, \dots, 10)$, 5 位检验位为 $p_i (i=1, 2, 3, 4, 5)$, 将检验码作如下安排:

检验码	p_5	a_{10}	a_9	a_8	a_7	a_6	a_5	a_4	a_3	a_2	p_3	a_1	p_2	p_1	
位序号	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
S_1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1
S_2	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0
S_3	0	1	1	1	0	0	0	0	1	1	1	1	0	0	0
S_4	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0
S_5	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0

可看出在上面虚线框内的每一列为位序号的二进制编码,实线框内的 5 行 15 列矩阵称为一致检验矩阵,矩阵中的每一行和每一列都是一个奇检验码。令:

$$S_1 \quad a_9 \ a_7 \ a_5 \ a_4 \ a_2 \ a_1 \ p_1$$

$$S_2 \quad a_{10} \ a_7 \ a_6 \ a_4 \ a_3 \ a_1 \ p_2$$

$$S_3 \quad a_{10} \ a_9 \ a_8 \ a_4 \ a_3 \ a_2 \ p_3$$

$$S_4 \quad a_{10} \ a_9 \ a_8 \ a_7 \ a_6 \ a_5 \ p_4$$

$$S_5 \quad p_5 \ a_8 \ a_6 \ a_5 \ a_3 \ a_2 \ a_1$$

均为奇检验码,则 $S_1 S_2 S_3 S_4 S_5$ 能反映 15 位汉明码的出错情况,如表 1 所示。

表 1 15 位汉明码的出错情况

S_1 S_2 S_3 S_4 S_5	出错情况
0 0 0 0 0	无错
$S_i (i=1, 2, 3, 4, 5)$ 中有一个为 1 其余为 0	p_i 错
$S_i (i=1, 2, 3, 4, 5)$ 中有两个为 1 其余为 0	汉明码中有两位错,但不能定位
1 1 0 0 1	a_1 错
1 0 1 0 1	a_2 错
0 1 1 0 1	a_3 错
1 1 1 0 0	a_4 错
1 0 0 1 1	a_5 错
0 1 0 1 1	a_6 错
1 1 0 1 0	a_7 错
0 0 1 1 1	a_8 错
1 0 1 1 0	a_9 错
0 1 1 1 0	a_{10} 错

(王玉祥)

cunchuqi guanli bujian

存储器管理部件 (memory management unit, MMU)

对内存寻址空间提供地址转换和保护,并支持虚拟存储管理和多任务管理的集成电路。MMU 与操作系统密切配合,实现存储分配以及内存与外存之间信息的自动调度,简化了操作系统对运行程序或进程的存储管理。

MMU 主要由地址合成器、地址分配表、屏蔽寄存器和写违章检测逻辑等组成,典型电路结构如图 1 所示。

存储器管理由段管理和页面管理组成。段和页面技术是实现虚拟存储的基础。段为每个程序提供

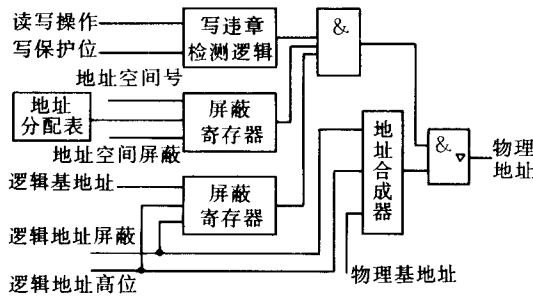


图1 MMU 电路结构

几个独立的、被保护的地址空间。页是利用磁盘存储器空间来模拟大量内存存储器空间。当几个不同的程序同时运行时,这两种方式都可以用来保护程序不受其他程序干扰。

对段的访问由段大小、访问特权级别以及段是否在内存存储器中等因素来控制。段管理硬件把段地址(逻辑地址)转换成连续的线性地址空间。如果在段中又使用页面技术,则页面管理硬件把线性地址变成物理地址;否则,线性地址作为物理地址使用。

页面通过数据在内存存储器和磁盘存储器之间的动态调度,实现访问大于内存存储器实际空间的数据。页面技术支持操作系统请求分页式虚拟存储管理方式。页面调度对用户是透明的,而分段则不透明。

典型的MMU产品有美国Motorola公司的MC 68451,它采用HMOS工艺。其主要性能有:将系统地址空间与用户资源分离,提供写保护,支持页面和分段技术,支持多MMU,允许通过共享段进行内部任务通信,简化程序地址空间的管理以及与直接存储器存取(DMA)兼容。

参考文献

1. Intel 486™ Microprocessor Family Programmer's Reference Manual. Intel, 1992
2. Bartee T C. Computer Architecture and Logic. McGraw-Hill Inc., 1991 (孙育宁)

cunchuqi leixing

存储器类型 (memory type) 存储器有多种分类方法。可按存储器在计算机中的作用分类,按存储器的存取方式分类,按存储媒体分类等。

按存储器在计算机中的作用分类,存储器可以分为**主存储器**(即内存存储器)、**辅助存储器**(即外存储器)、**缓冲存储器**等。主存储器用来存放计算机

运行时随时需要使用的程序和数据,它的工作速度较快,存储容量较小,主要采用半导体存储器,按随机存取方式工作。辅助存储器是一种不直接向中央处理器提供程序和数据的大容量存储器,它的工作速度慢,存储容量大,主要采用磁表面存储器和光存储器,按串行存取方式工作。缓冲存储器是位于两个不同工作速度的部件之间的、起缓冲作用的存储器。例如**高速缓冲存储器**、**先进先出缓冲器(FIFO)**等。

按存储器的存取方式分类时,存储器可以分为**随机存取存储器RAM**(也称读写存储器)、**只读存储器ROM**、**串行访问存储器**等。随机存取存储器是一种根据随机给定的地址对存储单元进行读写的存储器。每一存储单元的存取时间都是一样的,和存储单元在存储器中的位置无关,也和上一次访问的存储单元的地址无关。通用计算机的主存储器都采用随机存取存储器。只读存储器是一种对其内容只能读出而不能写入的存储器。其内容预先一次写入,用来存放固定不变的程序、汉字字型库、图形、符号等。由于它和随机存取存储器分享主存储器的地址空间,它也是主存储器的一部分。只读存储器还可用作微程序控制器中的控制存储器。**串行访问存储器**是一种对信息只能顺序地访问的存储器。信息存取时间和信息在存储器中的位置有关。辅助存储器采用串行访问存储器。串行存取存储器可分为**顺序存取存储器**和**直接存取存储器**。在顺序存取存储器中,信息完全按顺序在存储媒体上读写,例如磁带存储器。在直接存取存储器中,可直接找到存储器的某个区,然后在这个区内顺序存取,例如磁盘存储器或光盘存储器。

按存储媒体分类,存储器可以分为**半导体存储器**、**磁表面存储器**、**光存储器**等。在半导体存储器中,广泛使用的是金属氧化物半导体存储器,即MOS存储器。随机存储器主要采用MOS存储器,其器件可分为**动态随机存取存储器芯片(DRAM)**、**静态随机存取存储器芯片(SRAM)**、**视频随机存取存储器芯片(VRAM)**等。半导体存储器还有**只读存储器芯片(ROM)**、**可编程只读存储器芯片(PROM)**、**可擦编程只读存储器芯片(EPROM)**、**电可擦编程只读存储器芯片(EEPROM)**、**快可擦编程只读存储器芯片(Flash EPROM)**。常用的磁表面存储器有**磁带存储器**和**磁盘存储器**。磁盘存储器可分为**硬磁盘存储器**和**软磁盘存储器**。光存储器可分为**只读型光(CD-ROM)存储器**、**一写多读(WORM)光存储器**和

可擦光存储器等。

(郑衍衡)

cunchuqi xingneng

存储器性能 (memory performance) 表示存储器的工作速度(即存取时间)和存储容量的技术指标。

存取时间是指存储器从接到读或写的命令起,到读写操作完成为止所需要的时间。主存储器通常用半导体存储器构成,它的存取时间又可细分成两个技术指标:①取数时间。指存储器接收到读出命令到代码缓冲寄存器达到稳定所需要的时间。②存取周期。指存储器完成一次完整存取操作所需要的时间,它是对存储器进行连续存取操作的最短时间间隔。辅助存储器的读写机构常常带有机械运动装置,它的存取时间由两部分组成:①寻找时间。指从存储器接收到读或写命令起,到读写头定位到指定地址所需要的时间。②数据传输时间。它由存取时数据的传输速率决定。

存储容量表示存储器可以容纳的信息量,通常用千字节(kB)、兆字节(MB)或十亿字节(GB)表示。

(郑衍衡)

cunchuqi zucheng

存储器组成 (memory organization) 有层次结构的存储系统的组成及各层次的存储器的功能。存储容量和存取时间是存储器的两个基本技术指标。中央处理器的高速运算要求存储器能在很短的时间内完成指令和数据的存取操作。由于高速存储器的价格昂贵,因此存储系统通常由**高速缓冲存储器**、**主存储器**和**辅助存储器**3个层次组成。高速缓冲存储器的存取时间最短,但容量最小。辅助存储器的存取时间最长,容量最大。在操作系统的管理之下,依赖于程序执行过程中的局域性特性,由这3个层次的存储器所组成的存储系统可以提供接近于高速缓冲存储器的速度和相当于辅助存储器的容量。

一个存储器可由多个存储体组成。为了提高存储器访问的速度,可交叉地对多个存储体进行存取,这种存储器称为**交叉存储器**。它按模 m 交错编址。设存储器有 m 个存储体,令为: M_0, M_1, \dots, M_{m-1} 。每个存储体的容量为 S ,则第 j 个存储体的编址形式为: $m \times i + j$,其中 $i = 0, 1, \dots, S-1; j = 0, 1, \dots, m-1$ 。例如 $m = 4$,则存储模块的编址序列为:

$$M_0: 0, 4, 8, \dots, 4i + 0, \dots$$

$$M_1: 1, 5, 9, \dots, 4i + 1, \dots$$

$$M_2: 2, 6, 10, \dots, 4i + 2, \dots$$

$$M_3: 3, 7, 11, \dots, 4i + 3, \dots$$

在理想的情况下,交叉存储器的访问频度可比单存储体存储器的高 m 倍。但交叉存取时,有可能发生存储体访问冲突。其次,交叉程度越高,总线上的并联负载就越重,会使传输延迟增加。另外,计算机在执行转移指令时,会使交叉存取的效率降低,转移频度越高,效率就越低。选择适当的 m (例如令 m 为素数)可以提高交叉存取的效率。

近年来随着微电子技术的不断进步,中央处理器的运算速度与主存储器的访问速度之间、主存储器的访问速度和辅助存储器的访问速度之间的差距越来越大,存储间距问题随之产生。存储间距的产生与扩大引起了存储器组成方式,尤其是存储器层次结构技术的进一步发展;比较重要的发展包括**片上高速缓存**和**多级缓存**等。片上高速缓存通过将高速缓存和中央处理器制作在同一芯片上来获得非常高速(纳秒级)的存储访问能力。多级缓存通过在中央处理器和主存储器之间、主存储器和辅助存储器之间引入多级缓冲功能来进一步填补中央处理器、主存储器和辅助存储器(如磁盘存储器)之间的速度差距,从而提高存储系统的整体性能。

(黄震春)

cunchu quyuwang

存储区域网 (storage area network, SAN)

将存储单元互连,为计算机提供数据存储服务的专用网络。它通过可伸缩的网络拓扑结构互连不同类型的存储设备与服务器,提供内部任意结点间的多路可选择的数据交换,并将存储管理集中在相对独立的区域内,实现最大限度的数据共享和优化管理,以及系统的无缝扩充。

由于信息量快速增加,许多应用从计算密集型向输入输出密集型转变,原有以服务器为中心的存储方式无法满足网络环境中用户数据请求对输入输出性能的要求。此外典型存储系统(如并行SCSI存储系统)存在着可扩展性差等问题。于是人们沿用了大型机专用输入输出系统中的存储网络概念,并利用高速串行的光纤通道(FC)接口技术,构造了SAN。SAN可以看作是服务器的专门负责存储的“后台”网络,处理面向块设备的输入输出操作,如图1所示。它也可看作是一种扩展了的共享存储总线,使存储设备不专属于某一特定的服务器。

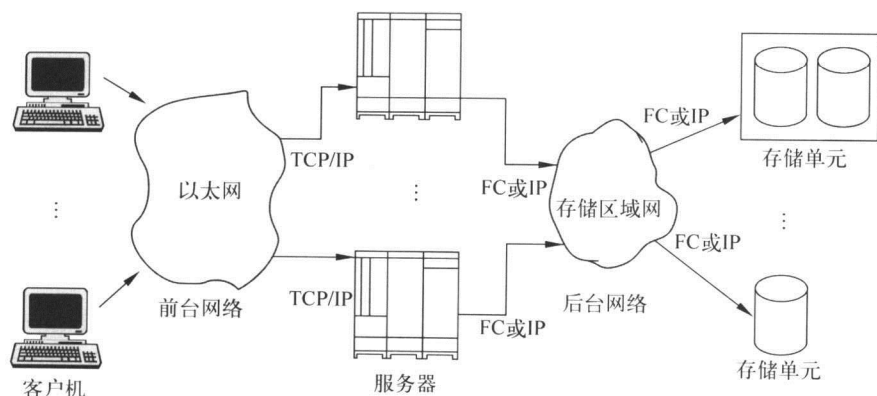


图1 SAN系统结构示意图

SAN 由一个提供物理互连的通信体系和一个管理层构成。但由于 SAN 概念的最早提出基于光纤通道技术的存储设备网络,并且市场上的产品大都是光纤通道存储区域网(FC SAN),使得人们误以为 SAN 即 FC SAN。事实上,其通信体系可以利用其他的串行 SCSI 技术,如 SSA,ESCON,HIPPI 以及各种网络。管理层负责对各个互连的存储设备进行组织,以确保数据传输的安全性和鲁棒性。

采用 SAN 的好处在于:逻辑上数据是一体的,管理上是集中控制的,结构上是易于扩充的。SAN 有较强的容错功能,可充分利用网络带宽提高存取速度,因而可以保证数据传输的可靠性,并提供高性能、高可扩展、结构灵活的存储服务。基于 SAN 可以很好地实现磁盘镜像、备份与恢复、备份数据的存档与检索、不同存储设备间的数据迁移以及网络内不同服务器间的数据共享。SAN 通常与其他计算资源如 IBM S/390 大型机相毗邻,但也可以利用广域网技术,如异步传输模式(ATM)或同步光纤网(SON)延伸到远程进行远程备份和数据存档。SAN 适用于多种存储密集型应用领域,如非线性编辑、服务器集群、远程灾难恢复、因特网数据服务等。

SAN 的物理拓扑结构随所用网络技术而异,性能也随网络性能差异而不同。以 FC SAN 为例,按拓扑结构 SAN 分为三类。

(1) 点到点 SAN(见图 2(a)) 具有网络远距离传输特性,能连接相隔 10 000 m 的远程存储设备,其远程连接能力不是传统总线所能具备的。连接配置中包括一发送端(光纤卡)和一目标端(可以是存储系统的光纤通道端口),无中继最远连接距离在

采用铜线连接端口时为 30 m,采用短波光信号连接方式时为 500 m。

(2) 环形 SAN(见图 2(b)) 是一种类似令牌环的共享带宽方式的拓扑结构。仲裁环路中至多可配置 126 个设备,设备间的通信通过仲裁以独占带宽方式进行。当环路中两设备间的通信完毕后才将控制权交给其他结点。每个结点都与其他结点共享一个单环带宽,双环路结构的带宽是单环带宽的两倍。环路连接一般通过光纤集线器来实现,环上的每个设备都会映射一个地址。当结点加入或从环路中移走时,环路会进入停止状态,并通过环初始化进程(LIP)重新分配相关结点的地址。环型 SAN 连接简单,但由于单存储设备故障将导致整个环路失效,系统安全性不高且不易管理。对于安全性和性能要求很高的应用,一般不适合采用光纤通道集线器连接方式。

(3) 交换式 SAN(见图 2(c)) 交换式 SAN 结构中存储结点通过 FC 交换机与其他结点进行一对一的通信,每对结点间的连接带宽为光纤通道带宽。理论上,交换式光纤网络可通过级联扩充方式容纳 1 600 万个结点。交换式 SAN 要求存储设备的光纤通道端口和光纤卡端口必须具有光纤登录能力。当端口登录到 FC 交换机时,经过信息交换,得到交换机分配给它的地址,同时登录信息将被注册到交换机中的单一名服务器表中。当结点加入和移出交换式 SAN 时,则不会产生环路初始化问题。如果两个结点间存在多于一个通道的连接,则当一个通道因故障不能进行通信时交换机将改变数据体的原始路径,以“路由”方式进行通信。

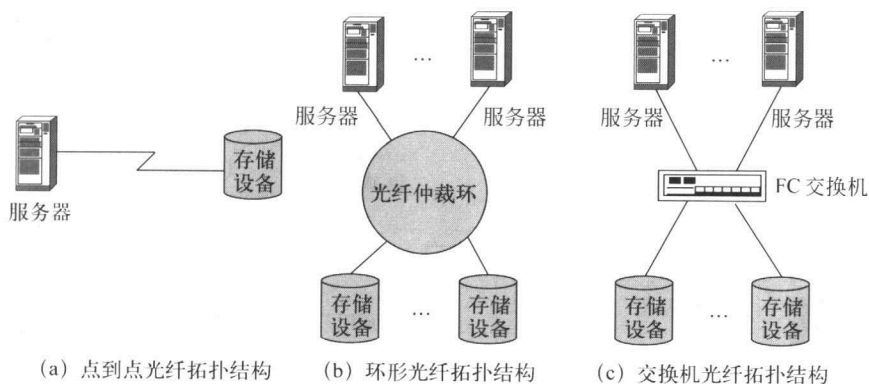


图2 SAN 的种类

SAN 的产品主要是 FC SAN, FC 具有高传输速率并且该 FC 网络只用于存储,独享带宽,因而数据传输速率稳定。但是,FC 设备价格昂贵,不同厂家产品兼容性差,需架设单独网络设施,运行维护成本高,又因 FC 网络不能运用一般网络传输协议,影响了它的普及和使用。与此同时,价格便宜的以太网技术飞速发展,并在以太网技术基础上出现了基于 IP 协议的 IP SAN。IP SAN 利用 TCP/IP 网络协议的易访问、易管理、技术较成熟等特点,在保持传输速率高且稳定的基础上,大幅度降低了成本。实现 IP SAN 有三种可选技术方案。

(1) iSCSI 因特网工程任务部 (IETF) 和存储网络产业协会 (SNIA) 共同支持的开放协议,它将 SCSI 数据用 IP 协议封装,使 SCSI 命令与数据能在 IP 网络上传输。

(2) FCIP 采用将 FC 数据封装在 IP 包中进行传输。由于数据经 FC 和 IP 两次封装,传输效率不高,但通过 FCIP 方式可以有效地与远距离 FC SAN 进行互连。

(3) iFCP 通过在 SAN 的终端设备上映射 IP 地址来引入 TCP/IP 协议。此协议要求 FC SAN 的终端结点上安装专门网关硬件,由该硬件将所管辖 SAN 发出的 FC 数据包进行 FC IP 协议转换和转发。iFCP 可以作为 SAN 内部协议或 SAN 间协议。

SAN 的带宽不断提高并且结构也越来越复杂。SAN 的低层互连技术如 FC 的速度从 2 Gb/s 向 10 Gb/s 发展,以太网从 1 Gb/s 向 10 Gb/s 甚至更高发展。IP 协议的引入可以较好地解决互操作性,但同时使 SAN 的拓扑结构越来越复杂,用户可以通过

主机、交换机、LAN/WAN 和 VPN 等设备访问存储数据。复杂的网络拓扑结构使存储网络面临着特殊的安全风险。在 SAN 的发展过程中,不仅要处理更快、更大、更方便的问题,还要处理更安全等新问题。

参考文献

1. 张江陵,冯丹. 海量信息存储. 北京:科学出版社,2003
2. Marc Farley. Building Storage Networks. USA: Osborne/McGraw-Hill,2000 (王芳)

cunchu xitong

存储系统 (memory system) 由计算机中的内存储器 and 外存储器组成的子系统。

现代计算机对存储系统有 3 个基本的要求,即存取时间短、存储容量大和价格(每一位的平均价格)低。这 3 个要求是互相制约的,存储器的存取时间越短,每一位的价格就越高;存储器的容量越大,存取时间就越长。根据所能达到的技术水平,仅用一种工艺技术做成的存储器系统不可能同时满足这 3 个基本要求。为此存储系统采用由小容量的**高速缓冲存储器**、**主存储器**和大容量的低速外存储器组成的层次结构,具有这种结构的存储系统称为**层次结构存储器**。

存储系统的层次结构如图 1 所示。图中从上往下,存取时间和容量依次增加,每位价格依次越少,即高速缓冲存储器的存取时间最短,容量最小,每位价格最贵;海量存储器的存取时间最长,容量最大,每位价格最便宜。在这种存储系统中,高速缓冲存储器的高速可以弥补主存储器在速度方面的不足,

而外存储器的大容量可以弥补主存储器在容量方面的不足,所以,具有层次结构的存储系统可以实现高速度和大容量,而且价格合理。

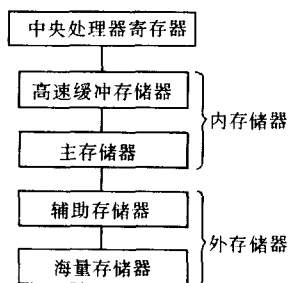


图1 存储系统的层次结构

采用层次结构的依据是存储器访问的局部性。所谓访问存储器的局部性是指中央处理器访问存储器时,在一定时间内,无论是存取指令或存取数据,所访问的存储单元都趋向于聚集在一个较小的连续单元区域中。例如,程序一般都包含若干个循环段,当某一个循环段运行时,中央处理器就反复访问这个循环段中的为数不多的指令。又如,在进行向量、数组、表格等操作时,中央处理器也只是对聚集在一块的数据进行访问。访存局部性分为时间上的局部性和空间上的局部性。时间上的局部性指的是最近的将来要用到的指令和数据可能是现在正在使用的。空间上的局部性指的是最近的将来要用到的指令和数据在存储器中的位置可能和现在正在使用的相邻或相近。根据访存局部性,将中央处理器在近期使用过的指令和数据区域存放在高速缓冲存储器中,就可达到中央处理器高速存取指令和数据的目的。在程序执行过程中,高速缓冲存储器中的内容要随着中央处理器存取的指令和数据区域的变化而改变,即高速缓冲存储器中的内容要和主存储器中的某些内容进行交换,这个交换是计算机自动完成的,对程序员透明。同样,主存储器的内容也可以成块地与辅助存储器进行交换。这样,从中央处理器的角度来看,层次结构存储器具有接近于高速缓冲存储器的速度,同时又具有接近于海量存储器的容量。

在访问这种层次结构的分级存储系统中的某一级时,如果所存取的内容已经在这一级中,称为命中,否则,称为不命中。如果不命中,存储系统自动到下一级存储器中去寻找,并把该内容所在的区域交换到这一级存储器中。如果所需要的内容也不在

下一级存储器中,则按同样方式到再下一级存储器中去寻找。提高命中率可减少各级存储器之间的数据交换,从而提高存储系统的效率。命中率和存储容量、所交换的数据块的映射策略、数据块的替换策略等有关(参见高速缓冲存储器)。

高速缓冲存储器中的指令或数据如果发生变化,主存储器中的副本应作相应的修改,使它们保持一致。这个问题在共享存储的并行计算机系统中变得更为复杂(参见高速缓冲存储器一致性)。

主存储器和辅助存储器之间的内容交换也是按块进行的。根据分块方式的不同,存储系统可分为页式存储系统(或称页式存储器)、段式存储系统(或称段式存储器)和段页式存储系统(或称段页式存储器)。在页式存储系统中,主存储器和辅助存储器都被划分为大小固定的页面,程序也被机械地划分为与页面大小相同的页,主存储器和辅助存储器之间的内容交换按页进行。在段式存储系统中,程序被分解为多个可以明确定义的段,它们相互独立或基本独立,但又在逻辑上形成整体。段的长短不是固定的。由于程序运行时所需的地址空间大小随程序而异,段式存储系统更能体现访存局部性,但给存储管理带来困难。段页式存储系统将页式存储系统和段式存储系统结合起来。在段页式存储系统中,将主存储器分为页面,将程序分为段,每个段又分为若干个和主存储器页面同样大小的页,以页为基本交换单位。主存储器和辅助存储器之间的页或段的交换有多种策略,常用的是将近期最少使用的页或段交换到辅助存储器中去。

(郑衍衡 孙强南)

cunchu zhuanfa jiaohuan

存储转发交换 (store and forward switching) 通信网络中的一种交换技术。在通信网的各交换机中设有缓冲存储器,由输入线路送来的数据先在缓冲存储器中暂存,等待输出线路空闲,必要时可对数据进行预处理。一旦输出线路有空,该数据就被转发到下一个交换机。这样,数据从发送端出发,经过一个个交换机的存储与转发,最后被传送到接收端。存储转发交换具有以下优点:①提高线路和交换机的使用率,并能实现流量控制;②必要时可对数据进行传送前的预处理,如格式转换、速率转换、分组的装拆,甚至增删、压缩等,从而增加了灵活性,扩大了功能,使不同数据传输速率,不同代码格式的数据站有可能相互交换数据;③配置适量容

量的缓冲存储器可以防止死锁、拥塞的发生；④能够进行路由选择，也可以把数据送到多个目的地；⑤易于进行差错控制和恢复；⑥能够进行数据的优先级控制。存储转发交换的缺点是传输延迟较长，不易满足实时的或快速交互会话的要求。

存储转发交换按存储转发的数据单位的大小可分为**报文交换**和**分组交换**。报文交换的特征是把要传送的数据不论其长短都作为一个单位——报文来进行存储转发；而分组交换则将报文分割成具有统

一格式，一定长度的报文分组，简称分组，并以此为单位进行存储转发。分组交换与报文交换相比，有传输延迟短、传输质量好、可靠性高、易实现多路通信和通信费用低等优点。

参考文献

1. 杜治龙. 分组交换工程. 北京：人民邮电出版社, 1993

2. 高星忠, 陈锦章, 张有材. 分组交换. 北京：人民邮电出版社, 1993
(史美林)

D

dayinji ceshi

打印机测试 (printer testing) 对打印机的质量与性能所进行的全面检测。打印机可按印字方式和印字处理技术分类。按照印字方式可分为串行式及并行式(行式),按照印字处理技术可分为击打式和非击打式。击打式打印机的主要代表是串行针式打印机,非击打式印刷机的主要代表有激光印刷机、喷墨印刷机等。

打印机测试主要包括打印速度测试、打印质量测试及打印功能测试。不同类型的打印机的测试方法不尽相同。

打印速度测试 打印机的速度可用计时器进行测量。串行针式打印机的打印速度有平均打印速度和打印速度之分,测试过程中需明确打印机的工作状态。平均打印速度是测试打印机在连续打满行(包括换行)时,单位时间内能打印的字符数,单位为字符每秒或汉字每秒;打印速度测量则是测试打印机在脱机状态下其出针第一列到最后一列的总时间除以一行内的字符数,单位同上。激光印刷机由于采用页处理方式,其打印速度通常以页每分表示,测试时只需记录每分钟打印的页数即可。喷墨印刷机的打印速度同样也以页每分表示,测试方法与激光印刷机相同,但在测试时必须注明打印覆盖率,覆盖率是指一定幅面的打印媒体上,打印面积占幅面总面积的百分比,通常在5%覆盖率下进行测试。

打印质量测试 测试针式打印机的打印质量,首先要测量打印精度,打印精度测试可用专用仪器检测连续打印的横线或竖线,通过对成行度、成列度、走纸累积误差的测量可对打印机的打印质量有定量的测试。激光印刷机和喷墨印刷机的印刷精度是以dpi表示的,即每英寸可印刷的最多点数。分辨率指标在打印机的设计中就已定型,测试时只需编制一段程序,考查打印机是否在一英寸里能打印出相应的点数,也可在放大镜下观察打印机可打印的最小点的直径,考查其是否与标称值相符。另外,打印质量测试中还需考查打印机的打印文本清晰度,通常可通过打印一段不同字体和字号的文字,观察其清晰程度来考核此指标。

打印功能测试 针式打印机功能测试包括复制份数、噪声、纸处理功能、打印特性、接口、耗材、自检等的测试。针式打印机的复制份数测试主要考察打印头的击打力度。噪声测试可反映打印机的综合表现能力。此外,不同的打印机的纸处理功能、接口、耗材、打印特性、自检等功能会有不同的表现形式,在测试时只需对打印机进行自检或联机操作,若均能按各种命令打印出符合打印方式与质量的打印结果,则认为打印机合格。激光印刷机和喷墨印刷机的功能测试与针式打印机基本相同,只是无针式打印机的复制功能,因此也无需再测试复制份数。

如打印机需在特殊的环境条件下使用,还应根据有关国家标准或军用标准对打印机进行振动、冲击、高低温、湿度、电磁兼容、电源适应能力、可靠性等方面的测试。

随着打印机技术的发展,其测试方法也在不断地变化。本文中未提及的其他类型的打印机如字模打印机、针式行式打印机、热敏式印刷机等,在具体的测试方法上可能会有不同,但总的来说可根据其类别参考上述的几个指标进行测试。(沈蓓)

daguimo bingxing chuli

大规模并行处理 (massively parallel processing, MPP) 采用由大量(数百至数万甚至更多)处理单元构成的并行计算机系统,处理单元之间通过相互通信和协作,从而快速、高效地对大型问题进行求解的过程和技术。

发展简史

大规模并行处理的历史可以追溯到20世纪60年代的ILLIAC IV,这是一台由64个处理单元构成的单指令流多数据流(SIMD)计算机。早期的MPP系统都是SIMD型的,基于多指令流多数据流(MIMD)机制的MPP系统的实际可用性一直受到学术界的怀疑。1983年,加州理工学院研制了一台由64个Intel 8086/8087组成的基于消息传递的多计算机系统,称为Cosmic Cube。这一系统随后发展为Intel iPSC/1和iPSC/2。nCUBE公司在1986年推出了10维超立方体系统nCUBE/ten,该系统最多可配

置 1 024 个处理单元。1988 年, Sandia 实验室在 nCUBE 上计算 3 个大规模科学工程问题, 得到了超过 1 000 倍的加速比。这一工作首次从理论和实践两个方面展示了大规模并行处理的现实意义。

在 20 世纪 80 年代, 在集中式共享存储 MPP 系统方面也进行了大量的研究和尝试, 如 IBM 公司的 RP3, 纽约大学的 Ultracomputer, 伊利诺依大学的 Cedar 等, 但硬件的复杂性和性能上的不足限制了它们的进一步发展。80 年代后期以来, 开始探讨在分布式存储器环境下实现共享存储的可能性。斯坦福大学的 DASH 就是一个典型的例子, 而 Kendall Square Research 公司的 KSP-1 则是第一个商品化的分布式共享存储器 MPP 系统。

体系结构

鉴于集中式的共享存储器系统受存储带宽的限制和在实现时的复杂性, 高度并行的计算机系统总是将物理存储器分散于各处理单元, 以便为每个处理单元提供足够的存储器带宽, 从而所有的 MIMD 型 MPP 系统在总体结构上都是相似的, 即由处理单元结点和互连网络组成。每个处理单元结点可以是一个由中央处理器、高速缓冲存储器、局部存储器和互连网络接口组成的单处理机(当然结点还可以有输入输出接口)。不同系统的差别主要体现在网络接口中。例如: ①早期的消息传递型多计算机系统采用存储转发(参见路由选择)的通信方式, 互连网络采用超立方体结构; ②后来的消息传递型系统在二维或三维网格上使用虫洞路由(参见路由选择)机制进行通信; ③在分布式共享存储系统中, 网络接口实现对远程数据的直接访问并维持数据的正确性; ④在基于高性能工作站或 PC 的网络计算环境(亦称集群式计算)中, 网络接口实际上就是网卡。

为了避免消息传递型带来的低效率和不方便, 并增强系统的通用性, 有必要提供虚拟的逻辑上统一的地址空间。因此, 出现了共享虚存和分布式共享存储器(参见分布式共享存储)的概念, 即在具有分布式存储器的系统中, 通过统一的虚拟地址空间来实现存储器共享。

完全由软件实现的共享虚存适合在消息传递型系统和基于高性能工作站的局域网上实现。此时, 每个处理单元的局部存储器被看作是整个虚拟地址空间的一部分, 地址转换类似于典型的虚拟存储系统。只是缺页中断的处理过程有所不同, 即新页面不但可能来自外存, 而且更有可能来自其他处理单元的局部存储器。与此相反, 在硬件实现的分布式

共享存储器系统中, 对非本地数据的访问是将远程数据由硬件自动取入本机高速缓冲存储器来实现。因为数据传输、共享和同步以及数据一致性操作的单位都缩小到了高速缓冲存储块, 这种方式具有比共享虚存方式高得多的系统效率。

程序设计与软件环境

并行计算机的应用程序开发主要有两种形式, 即串行程序的自动并行化和基于并行语言的并行程序设计。自动并行化的目标在于开发循环迭代间的并行性, 虽已取得了一些进展, 但效果仍不理想, 且主要针对 FORTRAN 语言进行, 适合于共享存储器的多处理机。基于并行语言的并行程序设计有多种方式, 如共享变量、消息传递、数据并行、面向对象、函数式、数据流式等, 但真正适合 MPP 计算机的主要是数据并行方式。

数据并行指的是将程序所要处理的数据域分割成许多小区域, 从而可用区域的划分来代替计算的划分。这样, 只要给每个处理机分配一个子区域, 整个计算就能完全并行地进行。适合数据并行的应用都具有这样的特点, 即对数据域中的每个元素都进行相同的计算, 所以这类应用也适合在单指令流多数据流(SIMD)系统上实现。

数据并行模型具有单线程、并行操作于不同数据、松散同步、全局命名空间、隐式通信和隐式数据分配等特点。

尽管数据并行方式在一定程度上简化了在消息传递型多计算机上编程的难度, 但仍然存在程序移植和并行化方面的困难。同时, 数据并行只是数据结构的一种形式, 且不能用于任务级的并行。消息传递方式编程时要为每个处理机单独编程, 非常复杂, 而且程序的伸缩性也不好。基于统一地址空间的共享存储器模型往往更适合于人们的思路, 随着分布式共享存储器多处理机系统的进一步发展, 基于共享变量的程序设计将逐步主导 MPP 系统的应用软件开发。虽然全新的非过程型语言在并行性的描述和开发方面有着 FORTRAN 语言和 C 语言无法比拟的优点, 但实现效率低, 并且大多和已有软件不兼容, 它们难以占领市场。

MPP 计算机的系统软件实际上从事着与单处理机系统软件类似的管理工作, 对大规模并行处理带来的新问题(如局部性、通信效率和负载平衡等问题)还缺乏有效的解决方案。纯粹依赖算法设计或程序设计来提高 MPP 系统的性能还不够。MPP 系统本身的并行化算法(即管理 MPP 计算机的算

法)将更直接关系到并行处理系统的工作效率。

模型和方法

并行计算模型是 MPP 体系结构和 MPP 算法之间的界面,在这一界面的约定下,并行系统的设计者可以设计对并行性的支持机制以提高系统的性能;算法设计者可以开发高效率的计算方法以充分利用并行系统的计算能力。一个成功的并行模型不宜对硬件和软件结构的细节作过多的限制,从而保证它在相当范围内的通用性,便于算法和程序的移植;但它又要能很好地反映出不同结构的主要特征。

程序设计模型在很大程度上体现了计算模型。例如,顺序计算已有很成功的计算模型,即随机存取(RAM)计算机模型,它与传统计算机的冯·诺依曼结构是直接对应的。RAM 模型的并行版本是并行随机存取(PRAM)模型,它实际上对应着共享存储的编程模型。它的优点是特别适合于并行算法的表达、分析和比较,使用简单,易于设计算法,稍加修改便可运行在不同的并行机上,且有可能在 PRAM 模型中加入一些诸如同步和通信等需要考虑的问题。但由于 PRAM 模型对存储器的访问时间、存储体的分布和存储体的访问冲突未作任何限制,不能直接指导基于共享存储模型的并行程序设计。与消息传递型程序设计方式对应的是通信顺序进程(CSP)模型。在这个模型中,多个并发的进程通过互相传递消息进行通信与协作,尽管 CSP 模型具有良好的数学背景,有利于并行程序的正确性验证,但它本身不提供任何解决诸如通信开销、网络延迟等问题的手段。

LogP 模型对基于消息传递的并行系统中的开销给予了较多的重视,它用少量参数 L, o, g 和 P 来刻画并行机的主要瓶颈,这个模型的详尽程度足以反映并行计算设计时的主要问题,其简洁性也足以支持详细的算法分析,从而可作为 CSP 的一个重要补充,但 MPP 系统的可编程性仍未很好地解决。数据流模型便于程序设计,且具有彻底的并行性,但具体实现和效率上的困难经过近 30 年的努力仍未克服。

研究并行计算方法的主要目的是开发应用问题计算中潜在的并行性,从而使这种计算便于高效地在并行计算机系统中进行。并行算法研究的目标是把计算的时间复杂性尽量转化为空间复杂性,基本做法是加宽算法树的宽度,压低算法树的高度。合理的排序与分解(如区域分解、算子分解、数据分级等)有助于负载平衡和减少通信量,这是大规模并

行计算提高效率的两个重要手段。

大量实践和分析表明,很多数学物理问题的潜在并行性体现在当规模增大时,并行加速比(参见计算机性能评价)呈线性增长。可伸缩性是研制并行算法的另一个重要指标。只有可伸缩的算法才能发挥可伸缩大规模并行计算机的高效率。适当的分块技术是实现可伸缩算法的重要手段,也是提高带有高速缓冲存储器的高性能计算机的性能所必需的技术。

存在的问题和发展方向

尽管 MPP 系统的峰值速度比向量多处理机高出 1 个数量级以上。但是 MPP 系统存在效率低、可编程性差、不同平台间并程序的移植难度大等问题,而传统的向量超级计算机经过多年的发展,已积累了大量高效率的应用软件。其系统软件,尤其是向量化编译程序已基本成熟。因此,许多实际计算问题在这类系统上可获得很高的系统效率。从系统所能达到的持续性能出发,并综合考虑花费在算法改进和程序编制方面的人力和时间因素,MPP 的性能价格比有时比向量超级计算机的还差一些。

造成 MPP 系统的峰值速度与实际速度相差很大的因素很多,如局部存储器带宽不够,通信速度与计算速度不匹配,输入输出性能不能满足要求,系统软件效率不高等。MPP 系统大都基于高性能微处理器,而大多数高性能微处理器即使在片内高速缓冲存储器完全命中的情况下每个时钟周期也只能进行 1 次存储器访问,这与它们每个周期能执行多条指令或完成多次浮点运算的峰值速度不相称。通信速度与计算速度的不匹配,主要原因不在于传输速度,而在于通信本身的开销。

理想的 MPP 计算机系统在各方面的性能都应该是匹配和均衡的。这些性能包括处理单元的标量和向量计算速度、处理单元存储器容量和带宽、处理单元的通信速度和 I/O 带宽、互联网络的通信带宽和网络中半数结点同另一半进行通信的最大带宽,以及整个系统的计算能力和 I/O 能力。其中,由于技术方面的原因,I/O 带宽与运算速度的匹配最难实现。

MPP 计算机都有大量并行的计算单元,但大都缺乏大量并行的输入输出单元,因而输入输出成为系统最大的瓶颈。早期的消息传递型系统把 I/O 工作都交给系统主机去完成,造成了严重的瓶颈现象,也增加了通信网络的开销。为部分或所有处理单元提供 I/O 能力和设备可以大大缓解这一瓶颈

问题,但由此带来的资源管理工作(如并行文件系统)仍是一个没有得到很好解决的问题。

传统单处理机迅猛发展的一个重要原因是其编译技术允许程序员用高级语言进行与机器结构无关的程序设计。相比之下,并行计算机(尤其是大规模并行计算机)的编译系统还远没有做到这一点。每个新的 MPP 机型的问世都面临着改写所有应用程序的艰巨任务。因此,为了进一步扩大 MPP 计算机的应用范围,必须实现与机器无关的 MPP 程序设计环境。

分布式共享存储模型可以方便地移植已有应用程序,便于开展自动并行化工作,因此具有一定的通用性。但是,它仍不能解决 MPP 系统实际速度远低于峰值速度的问题。相反,由于系统软件和优化技术在增强局部性、降低通信量等方面缺乏有效的手段,直接基于消息传递方式编程更利于发挥系统性能。消息传递和分布式共享存储模型都有一个不断完善的过程。

对于并行处理系统,下述几个部分是决定整个系统性能的关键:高速的单元处理机、存储系统、通信机制、同步机制和输入输出(I/O)系统。为了提高 MPP 系统的实际速度,一方面要改进这几个部分的设计,另一方面必须改善系统软件的效率,加强静态和动态优化的能力。例如,设法提高程序的访存局部性,有效地利用预取和后存技术,它们对提高 MPP 系统的性能有很大帮助。所有这些都需要体系结构、系统软件和算法设计三个方面的紧密配合。

参考文献

1. Gordon Bell. Scalable, Parallel Computers: Alternatives, Issues, and Challenges. International Journal of Parallel Programming, 1994, 22 (1): 3 ~ 46
2. Kai Hwang and Zhiwei Xu. Scalable Parallel Computers: Technology, Architecture, Programming. New York: McGraw-Hill, 1998 (唐志敏)

daxing jisuanji

大型计算机 (large-scale computer, main-frame) 使用所在时代的先进技术构成的一类高性能、大容量通用计算机。它代表该时期计算机技术的综合水平。

大型计算机的处理机系统可以是单处理机、多处理机或多个子系统的复合体。处理机一般采用两级高速缓冲存储器、流水线技术和多执行部件以提

高性能。存储器系统一般由高速缓冲存储器、主存储器、磁盘存储器和海量存储器组成,它们构成多层次的存储器系统。输入输出系统由通道和外围设备组成。通道(参见**输入输出通道**)一般有字节多路通道、选择通道、成组多路通道等。

发展简史

大型计算机的发展大致可以划分为 4 个阶段:第一阶段为 20 世纪 60 年代。这个阶段的典型代表是 1964 年美国 IBM 公司推出的 System /360 系列。它具有以下主要特征:其体系结构既便于事务处理,又便于科学计算;系列中各机型具有兼容性(参见**系统兼容性**);具有标准的输入输出接口,使输入输出设备与中央处理器独立,并提供在 System /360 所有机型中均能使用的输入输出设备;为保护用户的软件投资,采用既保持兼容性又可扩充的体系结构设计。System /360 属于采用集成电路的第三代计算机。在 System /360 推出之后,世界上许多国家的大型通用计算机公司相继开发与 System /360 兼容的系列机。第二阶段为 70 年代,这个阶段的典型代表是 1970 年 IBM 公司宣布的 System /370 体系结构。System /370 扩充了指令集,引入虚拟存储器和成组多路通道技术。第三阶段为 80 年代。典型代表是 1981 年 IBM 公司宣布的 System /370-XA。它采用扩充体系结构。地址空间由 24 b 扩大到 31 b。采用了浮动通道子系统。1988 年 IBM 公司又宣布了 ESA /370 体系结构,引入数据空间(数据专用的虚拟空间)和使输入输出高速化的超空间。第四阶段为 90 年代。以 1990 年 IBM 公司宣布的 IBM System /390 为典型代表。ESA /390 体系结构的特点是在 ESA /370 基础上增加多子系统复合体 sysplex, ESCON 通道体系和共同密码体系 CCA。

应用

大型计算机有十分广阔的应用领域。例如在军事领域,可用于全球性或区域性的战略防御体系、大型预警系统、航天测控系统等;在民用领域,可用于大型商用事务处理系统、全球和大区域中长期天气预报、云图处理和气象信息处理系统、大面积物探信息和资料处理系统、科学计算、大型工程设计和模拟系统等。大型计算机的设计和制造能力以及安装台数在一定程度上体现一个国家的综合国力。

发展趋势

大型计算机的发展趋势是采用 CMOS 微处理器构成多处理机或机群系统,即在保持用户应用程序兼容的前提下,利用 CMOS 微处理器大幅度降低价

格,而通过采用多处理机与机群技术,进一步提高性能。另一个发展趋势是开放系统。早在 1974 年,IBM 公司就宣布了网络体系结构 SNA。接着在 1987 年又宣布了应用体系结构 SAA。富士通、日立、NEC 公司也相继宣布了类似的体系结构。1991 年以后,各大型计算机公司又纷纷宣布信息系统构造新概念。如 IBM 公司的 Open Vision,富士通公司的 MESSAGE 90 s,日立公司的 FOREFRONT,NEC 公司的 Solution 21 等。开放系统的主要目的是使自己公司的大型计算机和个人计算机、工作站结合成用户需要的企业信息系统。系统也可采用其他公司的硬件和软件产品,如其他公司的 PC 机、工作站和独立软件厂商的 DBMS 和套装软件等,以显示大型计算机的作用是企业信息处理的中心和数据库、数据通信的中枢。

参考文献

1. Prasad N S. IBM Mainframe — Architecture and Design. McGraw-Hill Inc., 1989
2. 栗田昭平. コンピュータの最先端がわかる本. 日刊工业新闻社, 1992 (李经纬)

daxing jisuanji dianyuan xitong

大型计算机电源系统 (power supply system for large-scale computer)

根据大型计算机对电源的容量、质量、控制、检测等要求,把众多的直流电源、交流电源设备用供配电装置、监控装置连接而成的供电整体。电源系统的可靠运行,是计算机稳定工作的必要条件。

电源系统的设备

静止变换式不间断电源 大型计算机担负着重要的控制、运算或管理任务,即使短暂的停机,也会造

成重大的损失。所用电源能够利用蓄电池的储能,解决停电问题,并提高供电质量。

低电压大电流直流电源 计算机逻辑电路需要低电压直流电源供电。不同的器件需要不同的电压。每种电压需要总电流为数百到数千安。所以计算机硬件系统必须分割为规模适当的模块,使每个模块需要的电流适度,便于供电。大型计算机采用两类直流电源:多相整流电源 (PRPS) 和大功率开关电源 (SMPS)。这两类电源具有输出功率大、效率高、体积小、可靠性高的优点。另外,为保证供电的安全性,大功率电源内部都设有过压、过流、过热保护。所以,一旦有异常情况出现,电源就会自动关断,对负载和电源本身进行保护。

交流配电柜 交流电经配电柜的分路开关接到各个直流电源的输入。配电柜中设置引出保护线的端子,还有检测和显示交流电压、电流、频率、功耗等参数的功能,而且当某一参数超出规定范围,就会发出声光报警,确保机房供电的安全性。

电源系统的构成 开关电源系统 (图 1) 和多相整流电源系统 (图 2), 是大型计算机采用的两种电源系统。在开关电源系统中, 50 Hz 的不间断电源 (UPS) 和配电柜对开关电源供电, 同时又对外围设备和前端机供电。在多相整流电源系统中, 为缩小整流电源的变压器和滤波器的体积, 交流供电配备了 400 Hz 的不间断电源和配电柜, 专对多相整流电源供电。因为 400 Hz 的不间断电源不能用电作为后备电源, 所以为保证供电的连续性, 400 Hz 的不间断电源必须有备用设备。比较两种系统, 显然后者比前者设备多、投资大、交流供电复杂。另外, 多相整流电源输出不稳压, 只适用于功耗基本恒定的 ECL 器件。但它具有线路简单, 调试维护方便, 可靠

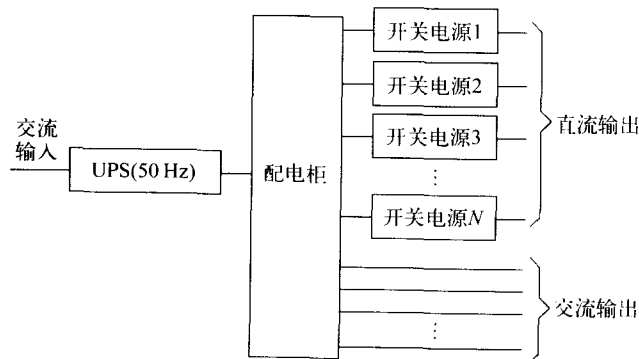


图 1 开关电源系统

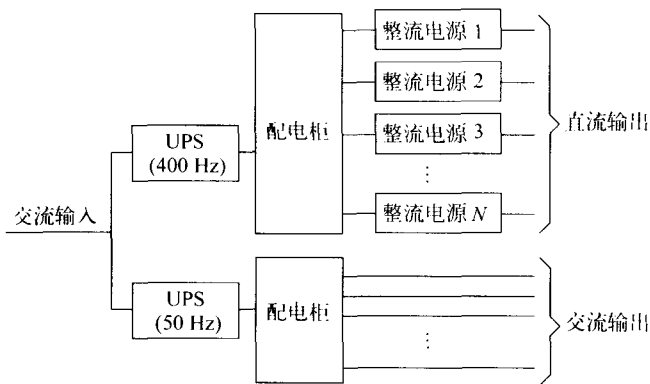
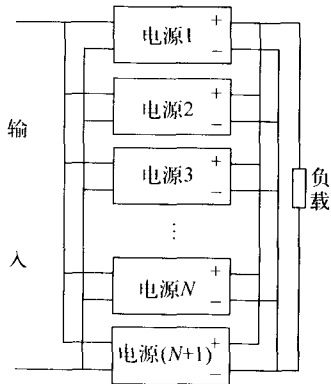


图2 多相整流电源系统

性特别高等优点。

电源系统的可靠性 大型计算机对电源可靠性的要求很高,通常采取以下技术措施:①交流电源采用两路市电供电,一路市电作为正常供电电源,另一路作为后备电源。这是提高交流供电可靠性的最有效,同时也是最经济的方法。②选用技术成熟的交流电源设备和直流电源。电源设备在实际使用中不能满负荷运行,要降低额定电流 30% 左右;并需改善环境条件,加强通风散热。③在特别重要的部位采用并联冗余结构。图3是 $N+1$ 冗余结构图,其中各电源性能完全一致,输出端允许直接并联,并能自动均流。负载电流可由 N 台电源提供,而实际电源为 $N+1$ 台,冗余 1 台。工作过程中,万一有 1 台电源出故障,系统仍能正常工作。在及时更换故障电源之后,电源系统仍处于冗余状态。

图3 $N+1$ 冗余结构图

电源系统的控制 大型计算机有很多电源,分

布在各个机柜内,多设有电源监控装置。监控装置通过遥控来开关各机柜的直流电源。为了减小开关过程中的浪涌电流,并观察加电过程是否正常,电源可根据程序安排的次序自动逐台开关,直到整个系统开机或关机过程结束。监控装置对各台电源的输出电压进行定期的采样和检测。发现电压异常,即将有关硬件脱机,使系统的其余部分照常工作;同时进行声光报警,指示发生故障的部位,以便维护人员进行处理。监控装置也对不间断电源的工作状态进行监测。如果发现交流市电故障,不间断电源已经进入蓄电池维持供电状态,就及时发出报警信号,以便进行应急处理。

参考文献

王其英. 计算机电源系统的设计. 北京: 科学出版社, 1987
(诸云龙)

daima shengcheng

代码生成 (code generation) 把经语法语义分析后的中间结果转换成等价的目标程序或目标程序模块的过程和描述。中间结果是用三元式、四元式或逆波兰式等中间语言表示的程序。目标程序是用目标语言书写的。目标语言可以是机器语言、汇编语言乃至高级语言。随着计算机的编译环境和执行环境的不断完善,不仅能产生可直接执行的目标程序,也能生成可再定位的或可连接的目标程序模块,并最终通过连接装入程序构成可直接执行的目标程序。

代码生成不但与编译有关,而且与运行环境有关。所生成的目标程序应当符合连接装入所要求的规范。代码生成中存储分配和寄存器分配是直接影

响目标程序功效的重要且复杂的问题,应给予足够重视。

参考文献

Aho A V, Sethi R, Ullman J D. Compilers Principles, Techniques and Tools. Addison-Wesley, 1986

(钱树人)

daima youhua

代码优化 (code optimization) 在语言处理过程中为提高程序质量而采用的技术。需求级、功能级和设计级语言处理系统的优化技术尚处于实验探索阶段,实现级语言处理系统尤其编译程序的优化技术发展较早、较为成熟。

编译阶段上可进行的优化分为中间代码级的优化和机器代码级的优化,又分别称为与机器无关的优化和与机器有关的优化。

中间代码级的优化有表达式优化、循环优化等。表达式优化通常有合并常量运算、提取公共子表达式、布尔表达式优化等。循环优化通常有外提循环不变式、强度削减等技术。

机器代码级的优化有寄存器优化、机器指令的有效利用等。

从涉及的范围上优化技术又可分为**局部优化**和**全局优化**。若优化涉及的范围是一个基本块,则称为局部优化,否则称为全局优化。

在优化工作中循环优化,尤其是与内循环有关的优化特别重要,它对程序的质量有显著影响。

优化技术的基础是数据流分析和控制流分析等。

目标程序的质量不仅与编译的优化技术有关,而且与源程序所采用的算法有关。求解一个问题的算法可能有許多,它们的复杂度常常会有数量级上的差异。采用高质量算法的源程序是提高目标程序质量的关键所在。

参考文献

Aho A V, Sethi R, Ullman J D. Compilers Principles, Techniques and Tools. Addison - Wesley, 1986

(钱树人)

daishu guiye

代数规约 (algebraic specification) 主要用于描述抽象数据类型的基于代数理论的形式规约方法。代数规约的数学基础是多类一阶等式逻辑。一

个代数规约由基调和公理集两部分组成。基调给出所涉及的类别,以及所定义的函数符号的类型。这些函数所应具有的性质则由一组公理来刻画。公理的形式通常是等式或条件等式。

例 自然数的代数规约

NAT =

sorts nat

opns 0: $\rightarrow \text{nat}$

S: $\text{nat} \rightarrow \text{nat}$

eqns $x + 0 = x$

$x + S(y) = S(x + y)$

$x * 0 = 0$

$x * S(y) = x + x * y$

对一代数规约的每个类别给定一个载体集,将函数符号解释为与其类型相应的集合上的函数,就得到一个代数结构。如果规约的每条公理在此结构中都成立,就说该结构是规约的一个模型。一般说来一代数规约可有无穷多个模型。比如单元素结构(载体集只有一个元素 e ,所有运算作用在 e 的结果均为 e),或者通常的自然数结构(载体集为 $\{0, 1, 2, \dots\}$,常量符号 0 解释为数 0,函数符号 S 解释为后继运算,eps, + 和 * 解释为通常的加法与乘法运算),都是上述例子中 NAT 的模型。NAT 还有无穷多个“非标准模型”。但是只要公理的形式是等式或条件等式(或更一般地,Horn 子句),则一定存在一个在同构意义下惟一的模型,从它到任一其他模型存在惟一的同态映射。这个模型称为该规约的初始模型。比如通常的自然数结构就是 NAT 的初始模型。采用初始模型作为代数规约的语义,就称为初始模型方法。

模块化的程序构造方式要求在已有规约的基础上引入新的类别和运算,即扩充式规约,或以满足某种要求的一类规约为参数,来定义新的规约,即带参规约。对于这些构造式的规约,除了初始模型语义外,还有终结模型或生成模型等不同的语义方法。

下面的例子给出了自然数上的“先进后出”栈的代数规约,这是一个扩充式规约。

例 自然数栈的代数规约

STACK_OF_NAT = **base** NAT

sorts nat stk

opns EMPTY: $\rightarrow \text{stk}$

PUSH: $\text{nat} \text{ stk} \rightarrow \text{stk}$

POP: $\text{stk} \rightarrow \text{stk}$

TOP: $\text{stk} \rightarrow \text{nat}$

ERROR: $\rightarrow \text{nat}$
 eqns POP(PUSH(x,s)) = s
 POP(EMPTY) = EMPTY
 TOP(PUSH(x,s)) = x
 TOP(EMPTY) = ERROR

代数规约用公理刻画程序的行为,抽象程度比较高。如何从给定的代数规约出发,得到能在计算机上执行的程序?这就是代数规约的实现问题。传统文献中关于这个问题的讨论多数局限于以低层规约实现高层规约,核心是实现的正确性。近年来人们开始探讨用传统高级语言中的模块实现代数规约,以及代数规约与前后断言规约的衔接问题。

如果将等式看成从左到右的重写规则,即可用其右边项替换左边项,一代数规约就成为项重写系统,从而可在计算机上直接执行,称为可执行的规约。

参考文献

1. Goguen J A et al. An Initial Algebra Approach to the Specification, Correctness, and Implementation of Abstract Data Types, Current Trends in Programming Methodology, Vol. 4. Data Structuring, R. Yeh ed. New York: Prentice-Hall, 1978
2. Ehrig H, Mahr B. Fundamentals of Algebraic Specification 1. Berlin: Springer-Verlag, 1985
3. Ehrig H, Mahr B. Fundamentals of Algebraic Specification 2. Berlin: Springer-Verlag, 1990

(林惠民)

daishu yuyi

代数语义 (algebraic semantics) 用代数结构描述的计算机语言的语义。它把计算机语言的语义定义为满足某种公理体系的抽象代数结构,并利用这种代数结构来研究用该语言编写的程序的模型论性质。

代数语义始于对抽象数据类型的研究。数据类型是计算机语言中的重要组成部分,但在 20 世纪 60 年代中期以前一直缺少科学的定义。它被认为仅仅是一些数据的集合,这种观点不能反映数据类型的内在数学特性,因而不能用来检验程序的正确性。1967 年问世的 SIMULA 67 语言第一次提出类型的概念,把数据和被允许施行于这些数据之上的运算结合为一个统一体,它是现代抽象数据类型的

开端,但当时未引起足够重视。20 世纪 70 年代初,软件危机促使人们去研究编写和验证正确的程序的理论和技术。在当时出现的一些新语言中,进一步把数据类型的特性与它的具体表示及实现方式分离开,提高了它的抽象程度。一个数据类型还可以规定对具有此类型的数据结构的表示和实现细节的屏蔽方式,包括对从数据结构外部访问其内部和从内部访问其外部的限制,由此出现了完整的抽象数据类型的概念。

基于抽象数据类型的思想,用代数结构描述数据类型的语法(包括类子和类子间的运算结构)称为基调,再用一组公理描述上述运算的推导规则。基调加上公理就成为代数语义学意义下的抽象数据类型,满足这组公理的一个模型即是该抽象数据类型的一种代数语义,称为 Σ 代数,其中 Σ 代表基调,因此又称基调代数。当一个计算机程序被看成是抽象数据类型时,该抽象数据类型的代数语义就是此计算机程序的代数语义。

有些人(如 C. A. R. Hoare)是在研究公理语义的背景下讨论抽象数据类型的,他们关心的只是某段程序的执行是否满足某组前后断言,其重点在抽象数据类型的证明论性质。这个方法不能说明在满足公理组的诸模型中,哪些是设计者想要的,哪些是设计者不想要或希望排除的,这些模型之间的关系又如何,等等。从 20 世纪 70 年代开始, Goguen, ADJ 小组和 Guttag 等人提出了以抽象数据类型的模型集合及其性质为研究对象,采用模型论和范畴论方法给出了程序的代数语义理论。这种理论在 20 世纪 80 年代又得到了很大的发展。

公理集的性质 稳健性和完备性是一组对偶概念。已知在齐性代数(只有 1 个类子)中一定有一个完备且健康的证明系统。可是对于非齐性代数(类子数大于 1), Goguen 和 Meseguer 发现了一个具有不健康公理系统的抽象数据类型。这表明,应该引进一组元公理,作为抽象数据类型中公理集必须满足的条件,才能保证得到所需的模型。

为了使抽象数据类型具有模块化和可重用性质,一般都首先设计较小的抽象数据类型,然后逐步扩充,形成从小到大的抽象数据类型体系,扩充时遇到的问题:小类型中原有的性质到大类型中会不会发生变化?这包括:小类型中原来不同的两个元素会不会由于大类型中新增加了一些等式公理而变成相等?小类型中原来属于某个类子的元素集会不会由于大类型中新增加了一些函数而膨胀起来?

(例如,如果小类型中只有整数加法,大类型增加了整数乘法,但没有说明 $2 \times 4 = 8$, 因此得到一个新的整数 2×4)。没有前一个问题的抽象数据类型称为是层次一致的,没有后一个问题的抽象数据类型称为是充分完备的。已知充分完备性是不可判定的,其他可判定性问题未完全解决。

模型集的结构 在同一抽象数据类型的两个 Σ 代数之间可以存在 Σ 同态关系。把 Σ 同态关系看成范畴论中的射,则同一抽象数据类型的所有 Σ 代数构成一个范畴。在某些条件下,例如当所有的公理都是 Σ 等式时,存在着在 Σ 同构意义下惟一的初始代数(初始模型)和终结代数(终结模型),采用初始模型作为语义的方法称为初始语义方法。如果只考虑有限生成模型,则当所有公理都是 Σ 等式时,同一抽象数据类型的全体有限生成 Σ 代数相对于该公理集取商得到一组模型,它们构成一个完全格。其中的最大元素和最小元素分别就是该抽象数据类型的初始模型和终结模型。

在一般情况下,终结模型是平凡的,即对每个类子只有一个代数元素与之相对应。有两种办法可以得到非平凡的终结模型。第一种办法是只考虑全体 Σ 代数的一个子集,在这个子集中寻找终结代数。第二种办法是在基层数据类型上建立扩充数据类型,然后在扩充数据类型中寻找终结模型。Wand 证明了,如果基层数据类型 B 相当“良好”地扩充为数据类型 E ,使得这个扩充是层次一致和充分完备的,则在 E 的所有有限生成模型中必定存在一个终结模型 M ,且 M 以 B 的初始模型为基子模型。Kamin 不需要层次一致和充分完备的条件,对任意的扩充数据类型 E ,他根据 Σ 同态的概念,提出了一种行为等价的判定法则,把 E 中的全体有限生成模型按其是否行为等价分成许多等价类,并证明了每个等价类都有终结模型。当扩充数据类型满足层次一致和充分完备的条件时,只有一个行为等价类,于是回到 Wand 的结果。这种方法也适用于带参数的抽象数据类型。

程序语言的代数语义 在形式上,代数语义都是以抽象数据类型为背景研究的。如果能把一个计算机程序看成某种抽象数据类型,则也就有了该程序的代数语义。由于一个程序可能对某些输入是无定义的,因此要把上述抽象数据类型扩充为偏抽象数据类型,相应的 Σ 代数称为偏 Σ 代数。这里一个项 t 可以是有定义或无定义的,也可以是在某些模型中有定义,而在另一些模型中无定义(参见 Σ (基

调)代数)。计算机程序的另一个特点是两个不同程序可能会有相同的执行效果,这意味着两个不同的模型可能实际上是同一个。对扩充类型来说,它体现在如下定义上:项 t_1 和 t_2 称为强等价,若在所有模型中都有 $t_1 = t_2$ 。它们称为弱等价,若把它们写成基层类型的项时,在所有模型中都等价。计算机程序的代数语义性质可分为三个层次。第一层:对只含变量、常量、加、减、乘和布尔运算的表达式语言,存在充分完备和层次一致的抽象数据类型。它是多模型的,所有模型构成一个完全格。它各项的强等价性和弱等价性都是可判定的。第二层:对动态结构和静态结构一致的命令式语言,如果只含赋值和条件语句,则它只比第一层少一个弱等价可判定性。含静态循环(循环次数在进入循环前确定)时也是这样。若还含动态循环,则相应的抽象数据类型是层次一致的,但非充分完备,而是弱充分完备。它是多模型的,但不构成完全格。第三层:对动态结构和静态结构不一致的命令式语言(例如含 goto 语句),可以建立相应的抽象数据类型,使第二层的性质在此还成立。它有一个弱初始模型和弱终结模型。它的全体极小模型构成一个完全格。

参考文献

1. 陆汝钤. 计算机语言的形式语义. 北京: 科学出版社, 1992
2. Bergstra J A, Heering J, Klint P. Algebraic Specification. New York: ACM Press, 1989

(陆汝钤)

danhui lu shuzi kongzhiqi

单回路数字控制器 (single loop digital controller)

一种以微处理器为计算、控制核心,配以相应软件,在外观及使用上类似常规模拟控制器的数字式控制仪表。又称单回路控制器。单回路控制器一般可接收多个输入信号,但只输出一个模拟量信号(有的型号可以输出多个信号),构成单回路直接数字控制。它可以由用户编制程序,组成各种调节规律,所以又称为“单回路可编程控制器”。

单回路控制器一般由微处理器、过程输入输出通道、正面板、侧面板、供电电源、数字通信系统等硬件部分和监控系统、基本算式编程系统等软件部分组成。

单回路控制器将控制中常用的比例积分微分(PID)、超前与滞后(E/L)、四则运算、开方等几十种算式写入只读存储器中。这些固化算式称为“软

件功能模块”。将这些功能模块根据用户的需要按某种规律“连接”起来,组成控制方案的过程,称为控制器的编程。编程工作通过专用的编程器或某些控制器本身所附的编程器进行。方法有“在线”编程和“离线”编程两种。“在线”编程要求控制器中的随机存取存储器 RAM 有较大的容量,并有可靠的掉电保护装置。

图1为单回路控制器的内部输入输出关系图,由此图可以看出,算式的处理部分可由用户从标准

算法库中任意选择并组态而成。

尽管单回路控制器的型号各不相同,但它们一般都具有以下共同特点:

(1) 数字量和模拟量显示混合使用,输入和输出信号采用国际统一标准的模拟信号 $4 \sim 20\text{mA DC}$ 及 $1 \sim 5\text{V DC}$ 。

(2) 外形结构、安装方式、正面操作面板的设置、操作及显示方式都与模拟控制器相似,使习惯于模拟控制器的人员容易掌握。

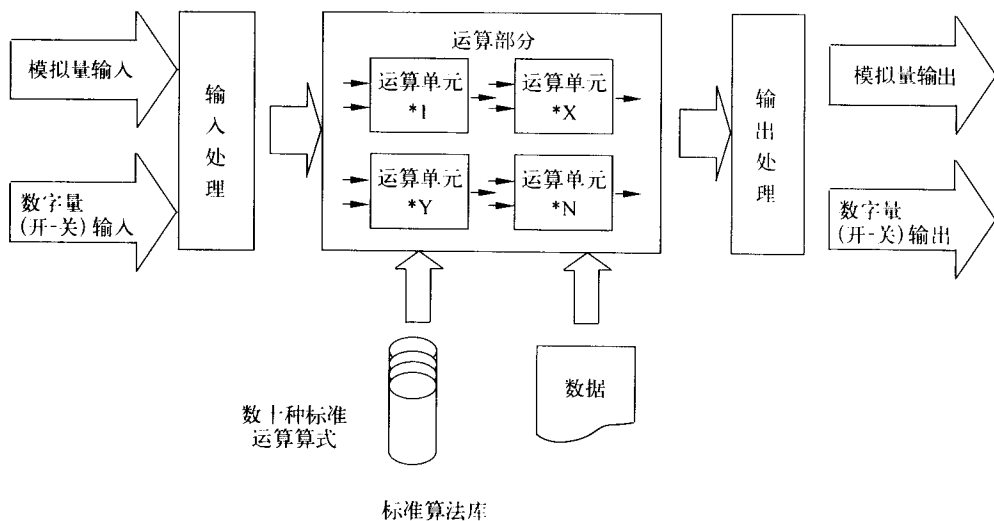


图1 单回路可编程控制器内部框图

(3) 用户程序编制采用“面向过程语言”,使用上类似袖珍计算器的编程。

(4) 控制器外部采用硬接线,与模拟控制器兼容;它的内部功能模块通过软件连接。控制器中配有数十种常用的控制算式和操作功能。用户可根据需要,按照系统的控制方案及算式,从中任意选择和组态。

(5) 具有数据通信功能。它既能代替模拟控制器单独使用,也可与其他单回路控制器或数字仪表、CRT 操作站、上位计算机进行信息交换,构成不同规模的计算机控制系统。

(6) 具有自诊断功能。能对仪表的各个功能模块和软件进行在线检查,发现异常立即显示诊断代码,指出故障部位并随时报警。

(7) 在仪表的软硬件开发上采用了后备操作、后备电源、无扰动切换、故障自动切换与隔离及冗余措施等可靠性技术。

从上述可知,单回路控制器并不仅是一种以取

代模拟控制器为目的的仪表,而是一台微型的过程控制专用计算机。现已广泛地用于冶金、化工、石油、电力等工业自动化中。

随着计算机、数字通信、人工智能、自动控制等技术的迅速发展,当前的单回路控制器正走向高度的智能化,并与现场总线技术等结合在一起,以满足不同工业过程的不同控制要求。

参考文献

1. 周春晖主编. 过程控制工程手册. 北京: 化学工业出版社, 1993
2. 黄桢地等. 过程控制仪表. 杭州: 浙江大学出版社, 1988

(王慧)

danpian jisuanji

单片计算机 (single-chip computer) 将中央处理器、存储器和输入输出接口集成在一个芯片上的微型计算机。简称单片机。由于单片机主要应用于控制系统,所以又称微控制器(MCU)。

20 世纪 70 年代中期在推出通用微处理器的同时,也出现了 4 位和 8 位单片机。如 National Semiconductor 公司的 4 位单片机 COP 400、Intel 公司的 8 位单片机 8048 和 Fairchild 公司的 8 位单片机 F8。70 年代末 80 年代初, Intel 公司、Motorola 公司和 Zilog 公司分别推出了与当时 8 位微处理器功能相当的 8 位单片机,从而使单片机得到广泛应用。80 年代以来,不但推出了 16 位单片机,并使广泛应用的 4 位及 8 位单片机功能更完善,增强了片内输入输出(I/O)功能,片内存储器容量也进一步增大。进入 90 年代后,推出了 32 位单片机。

单片机一般都采用面向控制的系统结构和指令系统。其中有的采用了程序存储空间和数据存储空间相互独立的哈佛结构,如 Intel 公司的 8051 单片机;有的则采用类 RISC 结构,如 Microchip 公司的 PIC 单片机。也有些单片机的中央处理器与某些微处理器兼容,如 Motorola 公司的 M 6801(M68HC11)单片机的中央处理器为 M 6800 微处理器。有许多 ASIC 电路的设计也采用单片机为内核,如 Intel 公司的 UC 51 中的内核为 80C51BH 单片机,它可以根据用户的要求定制 I/O 及存储器。

单片机的片内存储器由片内 RAM 和片内 ROM 或 EPROM(参见可擦编程只读存储器芯片)或 EEPROM(参见电可擦编程只读存储器芯片)组成。片内 RAM 较小,一般只有 64 B~256 B,用作通用寄存器、堆栈或数据存储器。片内程序存储器有片内掩膜 ROM、片内 EPROM 或 EEPROM 等形式。根据应用批量的大小,可以选用上述几种形式之一。一次编程的 EPROM 型单片机介于片内掩膜 ROM 与片内 EPROM 之间,较适合小批量生产。

单片机的多功能 I/O 结构是单片机的显著特点之一。较常见的通用 I/O 有定时计数器、并行 I/O 接口、串行 I/O 接口、数模转换器、模数转换器和 DMA 传送等;较常见的专用 I/O 有发光二极管、液晶显示器等的驱动电路、实时时钟、锁相电路、双音多频和声音合成电路等。有许多单片机带有串行数据通路,只需 2 根至 4 根 I/O 线即可方便地外扩 I/O 及存储器。如 Phillips 公司的 I²C 总线;Motorola 公司的 SPI 接口;National Semiconductor 公司的 Microwire 总线和 NEC 公司的 SBI 接口等。

单片机的应用大致上可以分为家用电器应用、一般控制应用、数据控制应用以及高技术控制应用。在家用电器应用中,较多采用 4 位单片机和低档 8 位单片机;在控制应用中,较多采用一般的 8 位单片

机;在数据控制应用中,较多采用高性能 8 位或 16 位单片机;在高技术控制应用中,较多采用高性能的 16 位单片机或 32 位单片机。

单片机的进一步发展有以下几个方面:①低电压、低功耗,可以在 1.2 V 电压下运行,电流仅为 μA 级;②高速执行和快速实时响应,不但采用了类 RISC(参见精简指令集计算机)的系统结构,并具有数字信号处理(参见数字信号处理器)的功能;③片内存储器容量进一步增大,可以把 FORTH, C 等高级语言和实时多任务执行软件进行固化;④片内 I/O 功能进一步增强,尽可能把应用所需的功能都集成在单片机内。

参考文献

1. 陈章龙主编. 实用单片机大全. 哈尔滨: 黑龙江科学技术出版社, 1989
2. Watson J D M. Microcontrollers. UK: Peter Peregrinus, 1990 (陈章龙)

dengzhimian gouzuo jishu

等值面构造技术(iso-surfaces construction technique) 三维空间数据场可视化的一种重要技术。这种技术可以利用现有的、由硬件实现的画面绘制功能构造比较清晰的三维空间数据场中的等值面图像,其图形生成及变换速度较快,因而被广泛地应用于科学及工程计算结果数据的显示中。

在三维空间数据场中构造等值面的方法很多,比较典型的是移动立方体方法。这一方法首先假定函数值在三维空间中均匀地分布在由立方体组成的三维网络的顶点上,并假定函数值沿立方体棱边作线性变化。当给定待求等值面的数值以后,首先需要判断等值面将与哪些立方体相交,当某一立方体 8 个顶点处的函数值均大于或均小于等值面的值时,则等值面将不与该立方体相交;否则,等值面将与该立方体相交。再根据函数值沿立方体棱边作线性变化的假设,求出等值面与立方体棱边的交点。将这些交点按一定规则连接起来,就可得到一系列的多边形或三角形。这就是待求等值面的近似表示。再利用计算机图形学中传统的画面绘制技术,就可以得到待求等值面的真实感图形了。这里要特别强调的是,在连接各交点时,要采用消除歧义性的算法,以保证连接的正确性。为了提高用多边形或三角形表示等值面的精确程度,也可以采用移动四面体方法,它是移动立方体方法的扩展。

移动立方体方法会将三维数据场中符合条件的

全部等值面构造出来。这在某些情况下是不必要的。解决的办法是采用由一系列二维轮廓线重构三维等值面。这就需要在一系列连续的二维图像中,进行图像分割,提取出感兴趣区域的轮廓线序列,然后再由一系列的二维轮廓线重构出三维等值面。

参考文献

1. Lorensen W E, Cline H E. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. Computer Graphics, 1987, 21(4): 163 ~ 169

2. 唐泽圣. 三维数据场可视化. 北京: 清华大学出版社, 1999 (唐泽圣)

di guidao weixing tongxing xitong

低轨道卫星通信系统 (low earth orbit communication system)

通信卫星的轨道定位高度为 500 ~ 1 500 km 左右的卫星通信系统。手持机个人通信通常利用低轨道卫星通信系统 (LEOS), 其优点在于: 一方面卫星的轨道高度低, 使得传输延时短、路径损耗小, 多个卫星组成的星座可实现真正的全球覆盖, 频率复用更有效; 另一方面, 蜂窝通信、多址、点波束、频率复用等技术的发展为 LEOS 提供了技术保障。

LEOS 具有以下主要特征: ①具有全球或区域

覆盖能力, 以适应未来个人化业务连接需要; ②利用极低轨道或网状覆盖倾斜轨道, 一方面可弥补同步轨道资源的不足, 另一方面又可支撑更优良的装备, 满足业务性能的需要; ③通信业务向多样化、综合化方向发展, 以期与未来多媒体高速信息传输相沟通; ④由于可能与全球个人业务相连接, 用户终端可使用类似或兼容于陆地蜂窝移动系统的蓄电池供电的小型手持机; ⑤系统设计及网络结构可提供进入或组合于现有公用通信网及陆地移动通信网的能力; ⑥网络设计、系统构成、星间协调、星上处理等充分利用现代通信智能化、数字化及多媒体化的最新技术, 以技术优势换取市场竞争和性能价格比上的优势; ⑦除轨道资源扩充外, 对频率资源亦进行积极扩充, 包括利用或混合利用 Ku、Ka 甚至 EHF 等高频段, 以及光频段开发, 以满足高吞吐量宽带业务传输及馈线链路和星际链路的构成需要。

低轨道卫星通信系统通常由卫星星座、关口地球站、系统控制中心、网络控制中心和用户单元组成, 如图 1 所示。在若干轨道平面上布置多颗卫星, 由通信链路将多个轨道平面上的卫星连接起来。整个星座如同结构上连成一体的大型平台, 在地球表面形成蜂窝状服务小区, 服务区内用户至少被一颗卫星覆盖, 用户可随时接入系统。

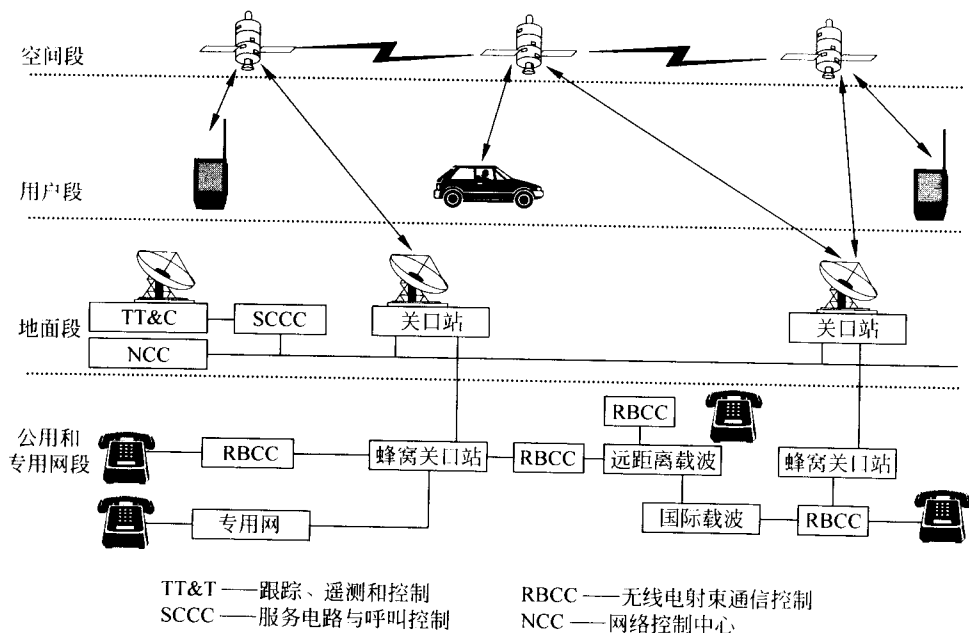


图 1 LEOS 的基本组成

最具代表性的 LEOS 主要有铱系统、全球星系统、白羊系统、低轨卫星系统、卫星通信网络、柯斯卡系统。

参考文献

1. 张乃通, 张中兆, 李英涛. 卫星移动通信系统. 北京: 电子工业出版社, 1997
2. 陈如明. 中、低轨道卫星通信. 第一讲中、低轨道卫星通信热及其基本特征. 电信科学, 1997, 13(7) (史美林 英春)

diji yuyan

低级语言 (low level language) 与特定计算机体系结构密切相关的程序设计语言。它包括字位码、机器语言和汇编语言。字位码是计算机惟一可直接理解的语言, 但由于它是一连串的字位, 复杂、繁琐、冗长、几乎无人直接使用。机器语言是表示成数码形式的机器基本指令集, 或者是操作码经过符号化的基本指令集。汇编语言是机器语言中地址部分符号化的结果, 其指令与计算机指令通常是一一对应的, 或进一步包括宏构造。用它们书写的程序不必经过翻译或只经过简单的翻译后就可以在计算机上执行。

低级语言的特点是与特定的机器有关, 功效高, 但使用复杂、繁琐、费时、易出差错, 程序员用数码形式或符号化的基本指令构造复杂的程序会由于涉及到太多的琐碎问题而工作量太大且易出错。通常使用低级语言进行程序设计目的是用它们可以写出执行速度更快且占用更小内存的程序。

参考文献

- Sammet J E. Programming Language: History and Fundamentals. Englewood Cliffs: Prentice Hall, 1969
(郑国梁)

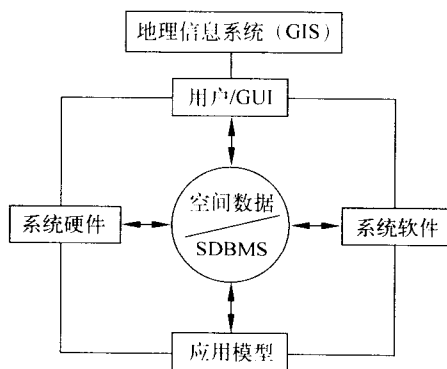
dili xinxi xitong

地理信息系统 (geographic information system, GIS) 为解决复杂的(自然)规划与管理问题而用于支持空间数据(指含有坐标、属性和拓扑关系的数据)采集、管理、处理、分析、建模和显示的一种信息系统。地理信息系统(GIS)的技术内容主要是管理和分析空间数据。它已形成了一门跨越地球科学、信息科学和空间科学的应用基础学科。GIS系统由计算机硬件、计算机软件和相关的方法、过程组成。

地理信息系统萌发于20世纪60年代初。1960年, 加拿大的 R. F. Tomlinson 提出了“把地图变成数字形式, 以便于计算机处理和分析”这一新思想。1965年, W. L. Garrison 正式提出“地理信息系统”这一术语, 并且沿用至今。

地理信息系统(GIS)主要研究在地理学理论依托下, 应用计算机技术对空间数据进行处理、储存、提取以及管理和分析过程中提出的一系列基本问题, 如空间实体的表达和建模、空间数据的获取和管理、空间数据的处理和转换、空间信息的查询和分析、GIS应用模型的建构、GIS的系统设计与评价、地理信息标准化研究、空间信息基础设施建设、GIS产品的可视化方法以及地学信息传输机理的不确定性(多解)与可预见性(多维)的研究等。它以地理实体的定位、定性和定量数据作为处理和操作的主要对象, 这是它区别于其他类型信息系统的主要标志。

地理信息系统(GIS)包括5个基本组成部分(图1), 其基本功能包含5大核心模块(图2)。



GUI——图形用户界面
SDBMS——空间数据库管理系统

图1 地理信息系统(GIS)的基本构成

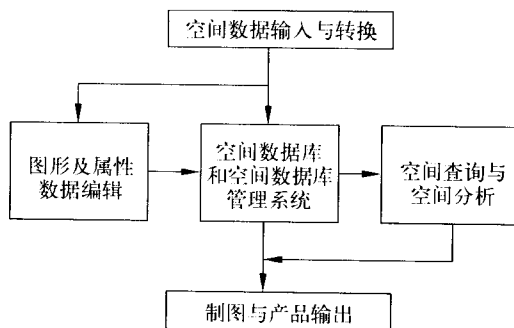


图2 地理信息系统(GIS)基本功能的主要模块

GIS 涉及的学科有地理学、测绘学、计算机科学、数学等。地理学广泛涉及人类居住的地球和地理空间,这与 GIS 的研究对象是一致的。测绘学及其分支学科,如地图学、摄影测量学等,不但为 GIS 提供定位数据,而且它们中的误差理论、地图投影理论和许多相关的算法等,可直接用于 GIS 空间数据的变换和处理。计算机科学为 GIS 提供了系统构成、软件设计、数据管理、图形生成和系统建立等的理论和方法。计算机领域的许多新技术,如面向对象方法、多媒体技术和虚拟现实技术、三维技术等,都与 GIS 的发展有着密切联系。数学的许多分支学科,包括几何学、拓扑学、分形理论等,已经广泛应用于 GIS 空间数据的分析、应用模型的构建和空间实体的表达。

数字地球和信息化浪潮的兴起给地理信息系统技术的发展提供了广阔的空间和机遇,地理信息系统技术正朝向集成化、产业化和全球化的方向发展。同时,由于地理信息系统具有空间规划、管理和决策等技术优势,可广泛应用于资源管理、国土监测、区域规划、市政工程、交通运输、企业决策和环保旅游等领域。

参考文献

1. 陈述彭主编. 地理系统科学. 北京: 中国科学技术出版社, 1998

2. 黄杏元, 马劲松, 汤勤. 地理信息系统概论 (修订版). 北京: 高等教育出版社, 2001 (黄杏元)

digui hanshu

递归函数 (recursive function) 一种以其早先的值来定义新值的可计算函数。

早在 1202 年, 意大利数学家 L. Fibonacci 提出兔子繁殖问题, 就开始了对递归的研究。假设兔子繁殖规则为: ①每对成熟兔子每月繁殖一对后代; ②兔子出生后第二个月成熟; ③老兔子不死。如果 1 月底引入一对刚出生的兔子, 问年底有多少对兔子?

用 $FIB(N)$ 表示第 N 月底兔子对数, 则 $FIB(N) =$

$$\begin{cases} 0, & \text{如果 } N=0 \\ 1, & \text{如果 } N=1 \\ FIB(N-1) + FIB(N-2), & \text{如果 } N \geq 2 \end{cases}$$

这个函数有一个突出的性质, 即用早先的值来定义新值。这就是递归定义。

所谓递归, 是将一较大问题归约到一个或多个

子问题的求解过程, 这些子问题在结构上与原问题相同, 但子问题求解比原问题简单一些。全面系统地研究递归函数, 应归功于 K. Gödel, 他在研究希尔伯特规划时, 发明并使用了原始递归函数的形式系统。

在定义递归函数之前, 先定义函数复合、递归、取极小三种运算:

复合 函数 $f(y_1, \dots, y_k), g_1(x_1, \dots, x_n), \dots, g_k(x_1, \dots, x_n)$ 的复合函数定义为

$$h(x_1, \dots, x_n) = f(g_1(x_1, \dots, x_n), \dots, g_k(x_1, \dots, x_n))$$

递归 函数 $f(x_1, \dots, x_n), g(x_1, \dots, x_n, y, z)$ 通过递归运算所得函数 $h(x_1, \dots, x_n, y)$ 由下列递归方程惟一确定:

$$\begin{cases} h(x_1, \dots, x_n, 0) = f(x_1, \dots, x_n) \\ h(x_1, \dots, x_n, y+1) = g(x_1, \dots, x_n, y, h(x_1, \dots, x_n, y)) \end{cases}$$

取极小 对函数 $f(x_1, \dots, x_n, y)$ 取极小, 得

$$g(x_1, \dots, x_n) = \mu y (f(x_1, \dots, x_n, y) = 0) \\ = \begin{cases} k, & \text{如果对所有 } z < k, f(x_1, \dots, x_n, z) \text{ 有定义} \\ & \text{且不为 } 0, \text{ 而 } f(x_1, \dots, x_n, k) = 0; \\ \text{无定义,} & \text{如果不存在这样的 } k \end{cases}$$

取极小算子通常称为 μ 算子。

部分递归函数类是含零函数 0, 后继函数 $x+1$, 投影函数 $U_i^{(n)}$, 并且在复合、递归、取极小运算下封闭的最小函数类。其中, 投影函数 $U_i^{(n)}(x_1, \dots, x_n) = x_i, n \geq 1, 1 \leq i \leq n$ 。

上述定义中, 若不含取极小运算, 则得到原始递归函数类。

递归函数类与图灵可计算函数类相同, 只要计算机能容许任意多和任意大的整数, 该类亦与由 PASCAL 或 FORTRAN 程序定义的整数函数类相同。无论是程序还是一般递归模式, 大多只能得到部分函数, 因为对某些初值计算可能不终止。

可计算函数属递归函数论的研究范畴, 递归函数论是数学领域的一活跃分支。计算机科学的实践与递归函数论有相当密切的联系, 递归的思想影响了程序设计语言的构造, 也影响计算机系统的结构。例如, 递归过程、递归数据结构、堆栈等都已成为计算机科学中的常用词汇了。

参考文献

1. Kleene S C 著. 元数学导论. 莫绍揆译. 北京: 科学出版社, 1985

2. Cutland N. Computability, An Introduction to Recursive Function Theory. Cambridge: Cambridge University Press, 1980 (陈火旺 贵可荣)

dianduidian lianjie xieyi

点对点连接协议 (point-to-point protocol, PPP) 在数据链路层中,为发送多协议数据报,通过一串点对点链路所提供的协议。

20 世纪 80 年代末,因特网工程特别工作组 (IETF) 推出了第一个 PPP 版本 (RFC 1134, 1989. 11)。当时只限于配合使用 TCP/IP 的路由器来应用。随着局域网上主机要求经过广域网发送报文日益增多,90 年代以来 PPP 发展很快。PPP 的主要任务是为所传报文提供一个标准的、可封装更高层次协议的机制,虽然此项封装是在数据链路层中完成的。

PPP 由 3 个主要部分组成:

- (1) 封装多协议数据报所用的方法;
- (2) 用于建立、配置和测试数据链路连接的**链路控制协议 (LCP)**;
- (3) 用于确定和配置不同的网络层协议的**网络控制协议 (NCP)**。

为了经过一串链路在两个对等层端点间全双工地按序传递报文,源发方可先进行 PPP 封装,即按 PPP 成帧要求组织好帧。然后,发出一个 **LCP 包**,一方面去配置和测试数据链路(如检查接收方是否支持 PPP 等);另一方面,在链路建成后,还需按 LCP 包的要求去协商一些可选功能(如最大包长度等)。接着,源发方还需发出一个 NCP 包去选择或配置一种或多种网络层协议。只有在配置好网络层协议后,才可经过这串链路去发送相应的信息包。这条链路可一直工作到 LCP 或 NCP 明确表示要关闭此链路时为止。

PPP 能运行于各种数据终端设备 (DTE) 和数据电路终接设备 (DCE) 接口,如 RS-232-C, RS-422, RS-423, V.35 等,其仅有的要求是需用全双工电路。

PPP 的帧结构经封装后如图 1 所示。

标志	地址	控制	协议	数据	帧检验序列
8 b	8 b	8 b	8 b 或 16 b		16 b 或 32 b

图 1 PPP 的帧结构

PPP 帧结构的各字段说明如下:

- (1) 标志字段,8 b 长,和 HDLC 的相同,为 7E

(16 进制),表示帧的起始或结束;

(2) 地址字段,8 b 长,为 FF(十六进制),表示 PPP 不编排个别的站地址;

(3) 控制字段,8 b 长,为 03(十六进制),表示仅用于传送用户数据;

(4) 协议字段,8 b 或 16 b 长,用来表明数据字段中所封装的是何种数据包(可能涉及 LCP, NCP, 安全性和压缩等)。如 C021(十六进制)表示 LCP, C023(十六进制)表示口令鉴别协议(PAP);

(5) 数据字段,不定长,它含有由协议字段指定协议的数据报,其最大长度用最大接收单元(MRU)表示,允许用户协商,默认值为 1 500 B;

(6) 帧检验序列字段,可为 16 b 或 32 b。

LCP 提供了建立、配置、保持和终止连接的方法,与之相应有 3 类 LCP 包:

(1) 链路配置包 用来建立和配置一个链路的包(包含 Configure-Request, Configure-Ack, Configure-Nak 和 Configure-Reject);

(2) 链路终止包 用来终止一个链路的包(包含 Terminate-Request, Terminate-Ack);

(3) 链路保持包 用来管理和排除错误链路的包(包含 Code-Reject, Protocol-Reject, Echo-Request, Echo-Reply 和 Discard-Request)。

LCP 包被封装在 PPP 帧的数据字段中,该帧的协议字段为 C021(16 进制)。LCP 包由代码、标识符、长度和数据 4 部分组成。

NCP 服务于更高层协议的连接。如用源发方某个路由器发送数据包(若采用互联网包交换(IPX)协议),它必须先确认接收方是否支持这个协议,还需协商建立此项连接的各参数值,以及在发送结束时及时拆链的方法等。因为用户环境涉及到各种网络层协议,故 NCP 有许多协议。

参考文献

1. Simpson W. The Point-to-Point Protocol (PPP), RFC 1661, July 1994

2. Thomas R. PPP Starts to Deliver on Interoperability Promises. Data Communications, April 1994, 23(4): 83 ~ 90

(黄令恭)

dianci jianrongxing

电磁兼容性 (electromagnetic compatibility, EMC) 电子设备(或系统)在共同的电磁环境中能一起执行各自功能的共存能力。该设备(或系统)不会由于受到处于同一电磁环境中的其他设备

(或系统)的电磁发射而导致或遭受不允许的降级,也不会使同一电磁环境中的其他设备(或系统)因受其电磁发射而导致或遭受不允许的降级。电磁环境包括设备(或系统)的辐射电磁环境和自然界中的电磁环境两大类。要求设备(或系统)在规定的电磁环境中按设计要求具有一定的抗干扰能力和不产生超过限度的电磁干扰,并能正常工作。

电磁干扰是人们早已发现的问题,1888年赫芝用实验证明各种打火系统向空间发射电磁干扰,1934年英国有关部门对一系列的干扰问题进行分析研究后发现,有50%的干扰是由电气设备引起的。为了加强对干扰问题的深入研究,国际电工委员会(IEC)成立了国际无线电干扰特别委员会,专门从事有关无线电干扰标准的研究和制定工作,其内容包括:保护无线电装置免受某些干扰的措施,干扰的测试方法及设备,干扰源产生干扰的允许值,电子设备的抗干扰度及其测试方法等。电磁干扰问题虽然由来已久,但电磁兼容这一新的学科却是近代才形成的。自20世纪80年代以来,世界各国在电磁兼容标准与规范、EMC设计与测量、EMC材料与工艺等方面进行了大量研究,使EMC的设计水平有了很大的提高。我国在1966年制定了第一个干扰标准:JB 851《船用电气设备工业无线电干扰端子电压测量方法和允许值》。80年代以来,先后又制定了30余项有关EMC的国家标准、国家军用标准和行业标准。这对推动我国EMC技术的发展和提高电子产品的可靠性,都起到了积极的保证作用。目前已有高精度的电磁干扰及其敏感度的自动测试系统,设备内及设备之间EMC的计算机辅助分析程序,从而形成了一套较完整的EMC设计体系。

电磁兼容性研究的基本内容包括:电磁干扰特性及其传播方式,电磁兼容性设计技术,电磁兼容性频谱利用,电磁兼容性材料与工艺,电磁兼容性测试和模拟技术,电磁兼容性规范与标准等。进行电磁兼容性设计时,应明确设备(或系统)在多强的电磁干扰环境中能正常工作以及本系统干扰其他系统的允许值;了解设备(或系统)的干扰源、被干扰源、干扰耦合途径等;根据具体要求,采取相应措施抑制干扰源,消除干扰耦合途径,提高电路的抗干扰能力。

电磁干扰源可分为自然干扰和人为干扰两种形式。自然干扰主要是雷雨、闪电产生的天电噪声,太阳黑子爆炸活动及银河系辐射产生的宇宙噪声等。人为干扰是由机电或其他人工装置产生的电磁干扰,如各种信号发射机、振荡器、电动机、开关、继电

器、机动车辆的点火系统、家用电器、高速逻辑电路、门电路、可控硅逆变器、整流器、照明设备、信息处理设备以及核爆炸时的核电磁脉冲等。随着科学技术的发展,人为干扰已成为电磁干扰的主要来源。为保证通信、广播与电视等电子设备正常运行,国际无线电干扰特别委员会制定了工业、科学、医用和家用电器及电动工具等人为干扰源的干扰极限值。我国在相应的国家标准及军用标准中也作了规定。这些标准有GB 4343《电动工具、家用电器和类似器具的无线电干扰特性测量方法和允许值》,GB 4824《工业、科学和医疗射频设备无线电干扰特性测量方法和允许值》,GJB 151《军用设备和分系统电磁发射和敏感度要求》等。

电磁干扰传播按其耦合途径可分为传导和辐射两种类型。沿电源线或信号线传输的电磁干扰称传导干扰,包括通过公共电源线直接传导,通过公共电源内阻耦合和经公共地线阻抗耦合而进入被干扰对象等几种形式。通过空间传播的电磁干扰称辐射干扰。干扰源中各种信号电路、电源电路导线在一定条件下可构成辐射天线,当干扰源外壳流过高频电流时,此外壳也将成为辐射天线向外辐射电磁能。干扰源周围空间可分为近场和远场两种区域,距离小于 $\lambda/2\pi$ (λ 为波长)的区域为近场区,反之为远场区。近场区内有电容耦合和电感耦合两种形式。远场区的电磁能量以电磁波的形式通过空间传播,作用到被干扰对象。

评定电磁兼容性能的指标是屏蔽效果。随着计算技术的发展,利用数值计算方法,可以对各种屏蔽结构的屏蔽效果进行定量分析和仿真模拟试验,其主要计算方法有基于传输和散射理论的屏蔽效果计算方法,基于并矢格林函数的孔缝耦合计算方法以及时域有限差分法等。

应用于电磁屏蔽的材料有衬垫材料、阻隔材料、屏蔽器件等。衬垫材料包括编制丝网、导电硅橡胶、金属丝纤维和网屏衬垫等,阻隔材料有屏蔽视窗、通风孔板、导电粘接剂等,屏蔽器件包括开关和转动部件的密封衬垫、箔带等。

提高电磁兼容性的具体措施有:屏蔽(电屏蔽、磁屏蔽和电磁屏蔽)、滤波、接地、限幅,恰当分配系统频率,正确选择连接电缆和布线方式,采用平衡差动电路、整形电路、积分电路和选通电路等技术。

参考文献

1. B E 凯瑟著,电磁兼容原理. 肖华庭等译. 北京:电子工业出版社,1985

2. 顾希如著. 电磁兼容原理规范和测试. 北京: 国防工业出版社, 1988 (赵悼爻)

dianhe ouhe qijian cunchuqi

电荷耦合器件存储器 (charge-coupled device memory)

利用具有电荷储存功能和电荷转移功能的电荷耦合器件 (CCD) 构成的串行存储器。电荷耦合器件是利用场效应管的结电容来保存电荷, 电荷的存储量大小取决于电极大小、形状和结构, 以及材料的势垒。电荷转移的过程则是利用两相或三相时钟进行控制。电荷耦合器件的输入部分由输入接触电阻、二极管和输入门组成。电荷通过输入接触电阻注入到第一个输入门的结电容上, 这个电路的充电时间常数远小于时钟周期, 因此在下一个时钟脉冲来到之前, 结电容能充电完毕, 其上的电压与电荷量成正比。在下一个时钟脉冲到来时, 利用二极管的反向特性将第一个门与输入电路隔开, 并将第一个门上的电荷转移到第二个门, 存储在第二个门。电荷耦合器件的输出部分由金属-半导体场效应管 (MESFET) 组成, 在时钟的控制下, 电荷从输入门转移到场效应管的控制栅上, 经放大后形成电压输出。

电荷耦合器件有直线型和平面型两种。直线型电荷耦合器件的电荷检测单元排成一行, 两侧是电荷转移单元, 如图 1 所示。平面型在原理上与直线型没有什么差别, 结构上不同的是电荷检测单元排成矩阵形式 (例如 100×100 或 240×100 的结构), 电荷转移单元则放在两行电荷检测单元之间。

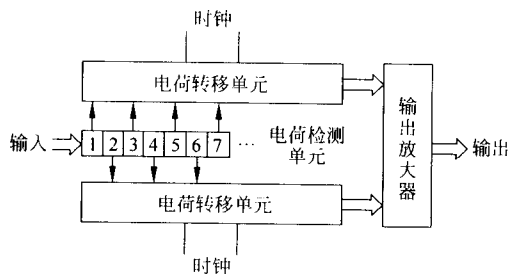


图 1 直线型电荷耦合器件结构原理

电荷的输入有两种方式。一是利用光照射电荷检测单元, 每个单元将接收到的光通量转换成电荷量。这样可将连续一片的光分解成一个个像素, 便于进行数字化处理。另一种输入方法是通过 PN 结将电荷注入到第一个电荷检测单元, 在时钟控制

下逐步转移到第二单元、第三单元。这样可将串行的电荷量转变为并行的电压量, 在时钟控制下输出。

电荷耦合器件功耗低, 尺寸小, 灵敏度高, 可靠性好, 抗震耐冲击, 因此很早就应用于摄像机和扫描仪中, 还用于光学符号识别和模式识别等图像信息处理设备中。 (黄德金)

diankeca biancheng zhidu cunchuqi xinpian

电可擦编程只读存储器芯片 (electrically erasable programmable read only memory chip)

一种存储内容可在电路系统中写入和擦除的半导体只读存储器芯片。简称 EEPROM 芯片。EEPROM 芯片可在电路系统中用电擦除, 不需从系统中取出, 也不需专门的紫外线擦除设备。EEPROM 芯片可像随机存取存储器芯片那样在系统中随机写数, 但写数周期时间长, 为毫秒级, 而且它允许写数的次数也是有限的, 但一般可大于 10 000 次。

早在 1970 年就开始使用金属-氮化物-氧化物硅 (MNOS) 栅区制作 P 沟道单元 EEPROM 芯片, 不久又制作出 N 沟道的。1984 年至 1985 年期间集成规模已发展到 64 kb, 由于减少功耗和设计专用逻辑的需求, 促成了 CMOS 工艺 EEPROM 芯片发展。以 64 kb EEPROM 芯片为例, 其静态功耗由 NMOS 时的 200 mW 降到 CMOS 时的 5 mW。EEPROM 芯片与单管结构的可擦编程只读存储器芯片 (EPROM 芯片) 不同, 它的存储单元由选择晶体管和存储晶体管组成, 双管单元结构的芯片功耗大、成本高, 限制了其集成规模的进一步发展, 到现在集成规模也只发展到单芯片 16 Mb。

EEPROM 芯片发展中一直伴随着特定的应用目标, 历史地形成了三个分别独立的产品段: 低集成、中集成和嵌入式。低集成指 8 kb 以下, 习惯上记为 EAROM 芯片 (电可修改只读存储器芯片), 主要用于收音机调谐器、汽车引擎控制器、销售点终端和邮资计量存储器、线驱动电话系统及工业自动化系统的刻度和参数设置。中集成 EEPROM 芯片产品系列是微处理机分布式系统、可换程序存储的需求所促成的, 还广泛用于自适应机器人、可编程视频图形发生器、可编程数据记录器和高速处理控制器等。嵌入式是指以嵌入处理器为目标的 EEPROM

芯片。与独立封装的电路相比,嵌入型具有较好的带宽、较少的接口电路和封装引脚,因之发展成一个独立的产品系列。嵌入式 EEPROM 芯片的特点是它有能力在内部产生写数据(编程)和擦除时所需的高电压。如 6805 微型计算机系列中 32 kb ROM 就使用了嵌入型 EEPROM 芯片,内含一个基于 3 ~ 6 V 可产生 18 V 写数据(编程)电压的电荷泵。嵌入型芯片已推广用于专用集成电路(ASIC)。

紫外线擦除的 EPROM 芯片可按位编程,但必须一次擦除全部信息,EEPROM 芯片则依不同的设计可选择字节擦除、页面擦除、体擦除等不同方式。图 1 为 32 kb EEPROM 芯片电路框图。该电路内有行地址和列地址控制,可选择 32 个字节组成的行、128 个字节组成的列和 32 × 128 字节的体,以实现不同的擦除选择。该电路还有透明的分区编程能力,允许通过熔断内部的多晶硅熔丝将其故障部分剔除,封装成 16 kb 或 8 kb EEPROM 芯片,以提高利用率。

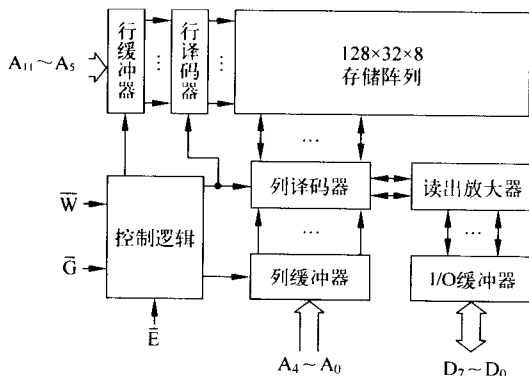


图 1 32 kb EEPROM 芯片电路框图

参考文献

- Rabaey J M. Digital Integrated Circuits: A Design Perspective. Prentice Hall, 1996. 北京:清华大学出版社, 1998(影印版) (时万春)

dianlan jieru jishu

电缆接入技术(cable access technologies)

通过电缆基础设施提供宽带接入的技术。电缆接入技术原本是为传输模拟视频信号而设计的,其中有三种基本技术用于宽带接入,它们是幅度调制 AM、频率调制 FM 以及数字技术。

由于有线电视系统基于同轴电缆模拟技术,因此不能为用户提供宽带服务所需要的任何额外的信

道或必需的频带。

从长远的观点看数字技术有很多优点:它不需要昂贵的遥测和监控系统,可提供比 AM、FM 系统好得多的性能,易于从同轴电缆过渡到基于光纤的系统,具有服务质量的保证,易于提供增值服务和信道,数字信号经压缩后能提供更多服务,从长远看更有效、更省钱。

在 20 世纪 90 年代中,人们探索用电缆基础设施来提供宽带接入的可能性,开发了电缆调制解调器。它可用于在混合光纤同轴电缆(HFC)网络中存取高速数据。从头端到电缆调制解调器的下行传输信道由多个电缆调制解调器共享,每个信道占下行信道频宽 6 MHz,转换成原始数据速率为 30 Mb/s。在上行方向,每个信道占上行信道频宽 600 KHz,转换成原始数据速率为 768 Kb/s。

有关电缆调制解调器的标准化工作由以下两个组织制定:一个是 802.14 工作组,它是 IEEE 的一个组织,制定了在传统有线电视网上的数据传输标准;另一个是由有线电视系统研究开发联盟开发的通过电缆服务接口传输数据规范(DOCSIS)。

参考文献

- 胡道元. 智能建筑计算机网络工程. 北京:清华大学出版社, 2002
- Padmanand warrier, Balaji Kumar, XDSL Architecture. McGraw-Hill Inc., 2000 (胡道元)

dianlu jiaohuan

电路交换(circuit switching) 通信网络中的一种交换技术。它在两个通信设备如电话或计算机之间经过几个交换结点创建一条直连的物理链路(电路),通信双方通过这条电路进行信息交换,如图 1 所示。

通过电路交换进行通信一般由电路建立、信息传输、电路断开三个阶段组成。例如在图 1 中 A 欲与 B 交换信息,则由 A 方向 B 方发起通信电路建立请求,通过各交换结点的路径选择、各交换开关的接续,建立起一条由 A 到 B 的物理连接, A、B 双方独占这条连接并进行信息(数据或语音等)传输。传输完毕拆除连接,电路断开, A、B 双方的一次通信结束。

电路交换网中的核心设备是各级的交换结点或称交换机。常用的电路交换机有空分交换机 SDS 和时分交换机 TDS 以及空分、时分相结合的交换机。空分交换机的基本原理是采用纵横开关矩阵及

多级纵横开关矩阵来形成输入输出交换路径,使得电路中的每一条路径在空间上与其他路径分开而互

不干扰。在时分交换机中,交换则是使用时分复用(TDM)和时间槽交换(TST)来实现的。

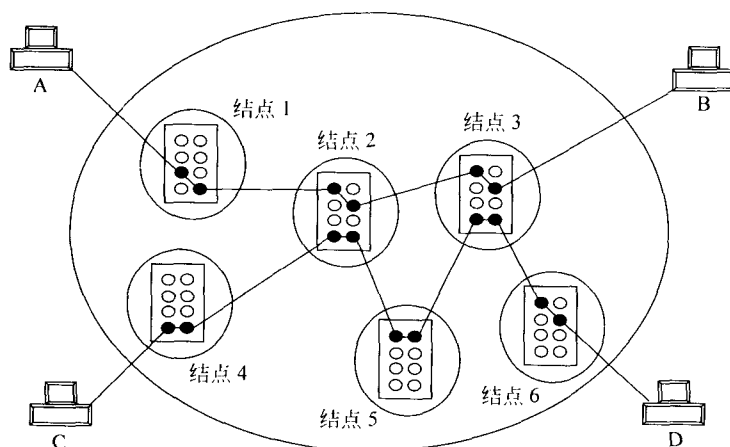


图1 电路交换

参考文献

1. Willian Stallings, Richard Van Slyke. Business Data Communications. 3rd edition. Prentice Hall (影印版). 北京: 清华大学出版社, 1997
2. Behrouz Forouzan etc. 数据通信与网络. 潘乞, 朱丹宇, 周正康译. 北京: 机械工业出版社, 2000 (史美林)

dianxin guanli wangluo

电信管理网络 (telecommunication management network, TMN) 国际电信联盟下的电信标准部 (ITU-T) 为了对电信网进行统一管理而提出的一种网络管理标准。国际电信联盟 ITU 在 1988 年就提出要制订一套对电信网进行管理的标准, 经过 ITU-T 下属第 4 工作组 (SG4) 几年的努力在 1992 年形成了 M. 3000 系列建议, 主要有: M. 3000 (电信管理网络 TMN 概述)、M. 3010 (TMN 的原理)、M. 3020 (TMN 接口规范方法学)、M. 3200 (TMN 管理服务)、M. 3400 (TMN 管理功能) 和 M. 3600 (ISDN 管理原理) 等。

电信管理网络 (TMN) 采用了开放系统互连 (OSI) 的网络管理框架, 但它用一个专门的**分组交换数据网 TMN** 来管理电信网。该专用分组交换数据网一方面连接电信网中称为网络元素 (NE) 的部分设备 (NE 实际上是被管的系统), 另一方面又连接了若干称之为运营系统 (OS) 的管理系统。网络管理员可以通过网管工作站及运营系统, 由专用分

组交换数据网收集和传输**网络管理数据**对形形色色的电信网络进行管理。电信网络可以是电话程控交换网、无线网或**综合业务数字网 (ISDN)** 等。电话管理网络 (TMN) 标准定义了各种资源互连之间的接口。如上述网络元素 (NE) 和运营系统 (OS) 之间的接口 (OS-NE 接口, 又称为 Q3 接口)。这些接口规范成为 TMN 的重要组成部分。

TMN 采用了面向对象的技术, 并正在向基于 CORBA 的方向演变。CORBA 是由对象管理组织 OMG 提出的, 已被 ISO 接受作为分布式对象的标准体系结构。由于 CORBA 有利于多厂商环境下的互操作性, 这正符合了电信网络管理的需要。

参考文献

- 谢希仁. 计算机网络 (第 2 版). 北京: 电子工业出版社, 1999 (高传善)

dianyuan ceshi

电源测试 (power supply testing) 对电源的电性能、安全性、电磁兼容性、可靠性等进行测量, 以确认被测电源的性能及其质量品质。

电源技术参数

(1) 输入端 输入电压 (交流或直流)、输入电流 (有效值)、输入电流正负峰值、输入功率、功率因数、效率。

(2) 输出端 直流输出电压、直流输出电流、输出功率、输出纹波 (峰峰值)、交流输出电压 (有效值)、交流输出电流 (有效值)、波形失真率。

测试项目

(1) 输入输出测试 测量电源在稳态状态下的输入及输出特性。

(2) 动态测试 测量电源在动态负载情况下的输出特性。

(3) 负载调整率测试 测量电源的一组特定的输出在三种不同负载情况下的响应,此时其他各组输出负载保持不变。

(4) 交互效应测试 测量电源的一组输出在其他各组输出负载变动情况下所产生的效应。

(5) 电压调整率测试 测量电源在三种输入电压情况下各组输出响应。

(6) 组合调整率测试 在三种不同输入电压及负载组合条件下,测量电源的输出特性。

(7) 开机及关机时序测试 测量电源在开机及关机瞬间各组输出的暂态响应。

(8) 输入叠加噪声干扰测试 检测电源抗干扰能力。

(9) 短路保护测试 当电源的一组输出对地短路后,测量其他各组的输出响应。

(10) 过载保护测试 在过载条件下,测量电源各组输出响应。

(11) 过压及电压过低保护测试 检测电源在输出电压过高或过低情况下的保护功能。

(12) 波形失真率测试 对直流变换成交流的器件或设备,测量电源输出中高次谐波所占的百分数。

(13) 耐压绝缘测试 测量电源输入对地和输出对地的绝缘性能。

(14) 机壳接地电阻测试 测量电源外壳的接地电阻是否满足相关规范的要求,以避免漏电及触电危险。

(15) 电磁兼容性测试 按照电磁兼容性的有关标准,检测电源满足电磁兼容性要求的能力。按照不同的使用要求,测试项目可包括雷击、浪涌、静电放电、电快速瞬变脉冲群、电流谐波、电压跌落、电压瞬变及短时中断、电压起伏和闪烁、辐射电磁场、传导干扰及辐射干扰等诸多方面的要求。

(16) 老化寿命测试 高温及长时间满载条件下对电源进行老化测试。

(17) 环境温度适应性测试 测量电源在规定的最高和最低工作温度情况下各组输出响应。

测试方法 早期电源测试是采用人工测试方法。测试者根据被测电源的类型、输出种类、技术要

求确定其测试项目,并依此利用必要的仪器仪表、设备器材构成测试回路,进行人工观测,测试结果容易受各种因素影响,客观性和一致性较差。后来,一些电源专业厂所应用电源测试系统进行自动测试。测试者根据所确定的测试项目编制测试程序,测试时测试系统在程序控制下实施对电源进行自动测试,显示并打印测试结果。测试快速、准确、高效、全面。

(张崇武 韩明)

dianyuan jicheng dianlu

电源集成电路(integrated circuits in power supply)

在直流电源和不间断电源中主要使用的集成电路。直流电源的工作过程是:将市电经过整流成为直流,然后通过脉宽调制(PWM)逆变器形成高压交流电,再整流稳压成直流电,供各种设备使用。不间断电源则是将市电经过整流成直流电,对电池充电,需要时通过脉宽调制逆变器形成交流电使用。这里,整流、脉宽调制、稳压都用到电源集成电路。常见的电源集成电路有整流桥堆、三端稳压器和脉宽调制控制器。

整流桥堆 将整流二极管按要求封装在一个壳体内。有半波整流、全波整流等品种。

三端稳压器 集成化的稳压线路。它有输入、输出和公共端三个端子。封装和三极管一样。三端稳压器除了有稳压作用外,一般还具有过电流限制保护能力,在结温超过额定值时具有自动截止电流的能力。具有体积小、使用方便的优点。因此,在微型计算机电源中应用非常广泛。

三端稳压器有固定式和可调式两种。78/79系列是常见的固定式三端稳压器。输出电压有5~24 V共9种,电流有100 mA,500 mA,1.5 A三种可选。其电压偏差范围一般为 $\pm 5\%$ 。

脉宽调制控制器 电源的核心部分。它向功率变换器驱动电路送出一串时间间隔相等的但脉宽不等的调制脉冲;或脉宽相等的、频率变化的调制脉冲。其控制规律是:当直流稳压电源的输出电压值偏高时,从PWM控制器送出的脉冲宽度变窄,或脉冲频率变低,从而将直流电源的输出电压幅度适当调低。反之,当输出直流电压偏低时,PWM控制器送出的脉冲宽度就变宽,或脉冲频率变高,使直流电源的输出电压适当回升。PWM控制器还对电源及负载提供过流保护和过压保护电路。

常用的脉宽调制控制器有两种类型:

(1) 自激式脉宽调制控制电路 由晶体管驱动

的高频变压器原边绕组和具有正反馈控制特性的高频变压器的副边正反馈绕组所共同构成的晶体管自激振荡器,不需要另外的振荡器和激励电路。这种脉宽调制控制器的调制脉冲的频率是变化的,但脉宽是不变的。

(2) 他激式脉宽调制控制电路 外加一个振荡器以产生开关脉冲,用来控制开关管的导通与截止。

有两种类型的专用组件可用为来构成 PWM 控制器:一种是电压型脉宽调制器,如 IR 3M02, KA 7500, TL 494, SG 3534, SG 3525, TDA 1060 等;另一种是电流型脉宽调制器,如 UC 3842, UC 3843, UC 3844, UC 3545 等。后一种脉宽调制控制器的脉冲周期保持不变,改变的是脉冲宽度。(林兼)

dianzi cidian

电子词典 (electronic dictionary) 内容存储在半导体存储器、磁盘、光盘等非纸介质上,可通过计算机对其查阅的词典。根据使用对象的不同,可分为两类。一类是供人使用的,仍用自然语言解说,但可提供灵活多变的检索手段,是书本词典不能比拟的。另一类是供机器翻译(MT)、人机会话等自然语言处理系统使用的,具有严格的形式化的表述方式,所记载的信息、知识都是代码化的。

存在于机器翻译系统中的电子词典是多种多样的。在直接方式的机器翻译系统中,只有一部双语对译的词典。在转换方式的机器翻译系统中,与分析、生成、转换三个阶段相适应,分别有源语言的分析词典、目标语言的生成词典以及两种语言对照的翻译与转换词典。在中间语言方式的机器翻译系统中,有描述概念及其关系的概念词典,也有从源语言到中间语言的分析词典和从中间语言到目标语言的生成词典,但没有两种自然语言对译的词典。以上词典所包括的都是自然语言的通用词汇及常规用法。为了翻译科学技术文献,则需要有各个学科与专业领域的专业词典。为了降低源语言分析的难度和提高目标语言生成的质量,惯用语词典与搭配词典也是需要的。上面介绍的分析词典、生成词典、双语对译词典、概念词典、专业词典、惯用语词典、搭配词典等都是系统词典,一般用户不能直接操作。除系统词典外,机器翻译系统通常还提供用户词典的框架,用户可以将自己常用的但在系统词典中尚未定义的词语及其语法、语义信息与用法填到这个框架中去。

电子出版物(包括电子词典)的市场日益繁荣,

电子词典在机器翻译系统中的重要性也日益增加。现在的发展倾向是系统中的语法规则力求概括,而词典中的语法、语义与用法信息却不厌其详。按照这种思路开发的机器翻译系统具有较高的可扩充性与稳健性。

参考文献

1. 冯志伟, 杨平. 自动翻译. 上海: 知识出版社, 1987
2. 陈肇雄主编. 机器翻译研究进展. 北京: 电子工业出版社, 1992 (俞士汶)

dianzi jisuanji

电子计算机 (electronic computer) 主要用电子器件和电路,配以各种相关设备而组成的自动信息处理装置或自动解算装置。根据信息或数据的表示方式和组成原理的不同,电子计算机大体上可分为电子数字计算机,简称数字计算机(参见**数字计算机**)和电子模拟计算机,简称模拟计算机(参见**模拟计算机**)两大类。这两类计算机的根本差别在于:①前者是对采用离散的符号或数字表示的信息或数据进行处理或运算;后者则是对用电流或电压等来表示的连续物理变量进行运算,以解算诸如解微分方程组等问题。②前者是在程序的控制下,顺序执行信息处理的操作,并可通过用户界面,实现人机通信;后者则事先在控制面板上排好解题程序,并设置好初始条件和有关参数,然后启动计算机,进行并行而连续的运算。③前者可以达到很高的精度;后者的精度则受器件和电源精度的制约。④前者主要用二值逻辑开关电路组成;后者主要用直流运算放大器连接而成。⑤前者有很强的通用性;后者的通用性差。

历史上的计算工具

早在远古时代,人类就有了数的概念和对数进行简单计算的方法,随后渐渐发展为简单的计算工具。我国在春秋战国时期(公元前8世纪到公元前3世纪),“算筹”已被用来作为计算工具,并已得到广泛应用。“算盘”在南北朝的著作(约公元570年)中已有记载,到元代(公元1271—1368年)已普及应用,而且沿用至今。应当提到的是:“二进制位”的概念起源于中国。《易经系辞》(约公元前3世纪初)中表示“阴”和“阳”的符号是“爻”。爻是一个只有两个元素的集合,也就是一个二进制位。易经中的八卦是3个爻的集合,六十四卦是6个爻的集合。德国数学家和哲学家 G. W. 莱布尼茨于

1679 年发表了关于二进制的论文,但他在研究易经后,于 1716 年的著作中说:“伏羲氏在其推演的八卦中使用了二进制算术。”还应提到的是“提花机”,这是一种织造提花织物的纺织机械。在织造前先将要织的花纹图案做成“花本”,然后用花本去控制织造过程。提花机是一种过程控制的纺织机械,花本就是存储的程序。我国在秦、汉时期已有提花机,长沙马王堆汉墓(公元前 2 世纪)中出土的提花织品已很精美。18 世纪末,法国用类似于穿孔卡片的穿孔纹板代替花本,自动织出各种花纹图案。

从 17 世纪开始,机械式的计算工具在欧洲得到了发展。例如:法国数学家和物理学家 B. 巴斯噶在 1642 年—1643 年间创造了一种用齿轮驱动的十进制加法器;莱布尼茨在 1674 年制作成功可进行十进制四则运算的手摇计算机等。英国的 C. 巴贝奇从 1822 年开始设计和制作“差分机”,这是一种可按事先规定的步骤计算多项式的机械式计算机,他在 1832 年制作了一个部件后,就转向设计可在穿孔卡片的控制下自动进行计算的“分析机”。由于种种原因,分析机未能制成。

从 19 世纪中叶到 20 世纪前期,有不少发明和创造性的工作,其中一些对后来电子计算机的发明和发展起着先导作用。例如 G. 布尔于 1854 年把逻辑归结为一种命题演算,提出了符号逻辑运算系统,并认为它可以作为计算机的设计基础,这就是后来获得广泛应用的布尔代数。在这个时期,计算机的发展从机械式转向机电式。1884 年第一个机电式加法机出现。1889 年建造出电气制表系统。1937 年,英国数学家 A. 图灵提出图灵机的数学模型。1938 年—1944 年,德国的 K. 宙瑟先后研制了 Z3、Z4 等二进制机电式过程控制数字计算机。1944 年—1947 年, H. 艾肯在美国哈佛大学研制成功用继电器组成的自动顺序控制计算机 MARK-I 和 MARK-II。

模拟计算工具的出现可以追溯到公元前数百年。代表性的模拟计算工具有 17 世纪 30 年代发明的圆周形计算尺和 1850 年的条形对数计算尺。对数计算尺适合于乘、除法和多种常用函数的计算,而且结构轻巧,便于携带,因此在随后的一百余年里,成为工程设计和科学实验人员的常备计算工具。19 世纪中期至 20 世纪 40 年代,也是模拟计算机从机械式到机电式的发展时期。例如,1930 年在美国 MIT 设计制作了机电式的微分分析机,用来求解各种微分方程。

电子计算机的诞生

1942 年,美国 J. V. 阿坦纳索夫等研制了一台二进制的电子管数字计算机原型,用于复杂而费时的理论物理计算。

1943 年,美国宾夕法尼亚大学莫尔学院的 J. P. 艾克特和 J. 莫奇里研制用电子管组成的电子数字计算机 ENIAC,该机于 1946 年 2 月投入运行,并公诸于世。当时它是一台前所未有的、复杂而庞大的电子机器,用了 18 000 多支电子管,重量达 30 t,机房面积约 140 m²,并达到了其他数字计算机望尘莫及的运算速度,即每秒运算 5 000 次 10 位十进制数的加、减法运算。ENIAC 开创了电子数字计算机蓬勃发展的新纪元。

1945 年—1946 年,美籍匈牙利数学家冯·诺依曼等提出了“存储程序”的概念和采用二进制逻辑与存储过程控制的电子数字计算机的基本设计思想和体系结构,即所谓的冯·诺依曼结构(参见**数字计算机**)。

从 20 世纪 50 年代初期开始,陆续推出了商品化的计算机。在 20 世纪后半半个世纪到 21 世纪,电子数字计算机得到了极大的发展,从最初的电子管计算机到其后的晶体管计算机、集成电路计算机和超大规模集成电路计算机。

在模拟计算机(参见**模拟计算机**)方面,自从 1952 年电子管的运算放大器投入市场后,电子模拟计算机及其应用开始迅速发展。早在电子数字计算机问世之前的第二次世界大战期间,模拟计算机就已经在武器的实时跟踪瞄准和射击控制等方面得到应用。此后又在某些领域,例如工程设计的动态模拟、生产过程控制、动态系统仿真等,获得了广泛应用。在 20 世纪 50 年代后期,出现了把数字计算机和模拟计算机结合起来组成混合计算机(参见**混合计算机**)。此外,还有一种用数字积分器来取代模拟计算机中运算放大器的数字微分分析机(参见**微分分析机**)。由于数字计算机的高度和广泛的普及应用,在许多传统上以模拟计算机为主的应用系统中,往往可以更好地用数字计算机来取代,所以通常所说的电子计算机或计算机指的就是电子数字计算机。

近代计算机的应用

电子计算机开始研制时,主要是用于科学和工程计算,但它的计算过程实质上是对二进制编码符号进行逻辑推理的过程。因此,它已经不仅仅是一种传统意义上的数学计算工具,而是在更广泛的意

义上具有一定智能属性、在过程控制下能自动运作的信息处理系统。由于计算机的通用性、高速、大容量以及不断提高的性能价格比,其应用已遍及现代人类社会的方方面面,特别是计算机和通信技术的结合使计算机的应用范围更为广阔。人类进入信息化社会主要是由于计算机的广泛使用。

计算机应用主要包括计算密集型、数据密集型和通信密集型3种类型。计算密集型应用包括大规模科学和工程计算以及数值模拟等,数据密集型应用包括数字化图书馆、数据仓库、计算结果可视化等,通信密集型应用包括计算机支持的协同工作、遥控、远程医疗等。

计算机应用领域举例如下:

(1) 生物学、生物化学和医学——药物设计、生物分子结构、基因信息学、医学图像处理、人体仿真、医学数据库分析、医疗管理系统等。

(2) 化学和化工——催化剂和酶的研究和设计、化工厂管道系统的设计等。

(3) 物理学——材料性质研究、芯片中的半导体模拟、结构力学计算和性能模拟、流体动力学计算、核聚变和核爆炸模拟、基本粒子性质研究等。

(4) 空间科学和天文学——宇宙形成研究、银河系形成研究、太阳动力学、黑洞和引力波研究、太空探测数据处理等。

(5) 人工智能——人工神经网络学习和优化算法、数据库检索、计算机下棋等。

(6) 天气和气象——天气预报和气象预测、灾害性天气预报等。

(7) 环境科学——污染物和地下水在地层中的流动模型、地球生态环境模拟、根据卫星遥测数据研究土地资源的性质和利用等。

(8) 地球物理和石油工程——石油勘探中的三维地震资料处理、油藏模拟、地震预报研究等。

(9) 航空、航天、机械和制造工业——汽车和飞行器的设计和制造、推进器设计、飞行器的雷达成像、计算机中的芯片模拟、高频电路的电磁特性模拟、生产过程模拟、制造系统的优化调度等。

(10) 军事应用——武器设计、军用传感器和通信系统模拟、军事决策支持系统、军事演习模拟等。

(11) 商业和事业系统——非正常情况(天气异常或人为事故)下的空中交通的动态调度、机器人的控制和图像分析、数字化电影中的图形学、经济模型、电子商务、税务系统、商业决策支持系统、银行业务支持系统、信用卡和股市信息支持系统等。

(12) 社会——数字电视、数码照相机、视频会议、远程教育、数字化图书馆、社会保障信息支持系统等。

展 望

电子计算机对经济发展、社会进步和科学研究将起到越来越重要的作用,它将使人类进入更高层次的信息化社会。高性能计算机的发展将使过去一些难以解决的极其复杂的重大问题获得解决。网格计算(参见**网格计算**)将分布在不同地理位置和不同类型的计算资源通过互连网络连接起来,实现资源共享和协同工作。用户使用网络资源时,不需要知道资源的提供者及其所在的地理位置。移动式计算(参见**移动式计算机**)可使用户通过互连网络在任何地点和任何时间进行计算。嵌入式计算机(参见**嵌入式计算机**)可直接装入机电设备、仪器仪表或家用电器内部,它们将在人们的周围环境中存在而不为使用者所觉察,它们无处不在,无时不在。

对于计算机所用的器件,自从使用半导体微电子技术以后,集成电路芯片的集成度(即单位面积所能集成的晶体管数量)基本上是按照摩尔定律增长的,即每隔18个月左右,集成电路的集成度增加一倍,其性能也增加一倍。然而在21世纪初期,硅半导体大规模集成电路技术已发展到接近于它的物理极限。为了进一步提高计算机的性能,除了在计算机体系结构方面将计算机组成高度并行和分布式的系统外,还将开发新型器件。例如纳米器件、量子器件、具有量子效应的超导器件、光调制器件、光电子集成器件、分子器件、生物分子器件等。利用它们有可能组成新型的高性能计算机或高性能协处理机。

总之,现代电子计算机历经半个多世纪的发展,已经成为当今文明社会须臾不可或缺的、最得力、最灵活、最普遍的信息处理工具。今后仍将继续沿着提高其性能、强化其功能、增加其易用性和灵活性、进一步拓广和深化其应用等方面不断发展。电子计算机的前景是不可限量的。

参考文献

1. Ralston A, and Edwin E D, ed.. Encyclopedia of Computer Science. 3rd ed.. N. Y.: IEEE Press, 1993
2. Timeline of Computing History. IEEE-CS: Computer, October, 1996
3. Saha D, and Mukherjee A. Pervasive Computing: A Paradigm for the 21st Century. IEEE-CS:

Computer, March, 2003, p. 25 ~ 31

(童颖)

dianzi shangwu

电子商务 (electronic commerce, EC) 利用信息与通信技术进行商务运作及与其有关的信息交换与管理的过程和技术。

电子商务涵盖的业务包括: 信息交换、销售、售前、售后服务(提供产品和服务的细节、产品使用技术指南、回答顾客意见)、电子支付(使用电子资金转账、信用卡、电子支票、电子现金)、运输管理(包括商品的发送管理和运输跟踪, 以及可以电子化传送的产品实际发送)、组建虚拟企业(组建一个物理上不存在的企业, 集中一批独立的中、小公司的权限, 提供比任何单独公司多得多的产品和服务)、提供公司和贸易伙伴可以共同拥有和运营共享的商业方法等。

电子商务是网络经济中企业商务运作的一种重要模式, 是多种技术的集合体。企业的商务活动通过计算机网络的整合, 特别是 **Internet**、**外联网** 和 **内联网** 的整合, 综合运用信息技术以协调和协作方式进行, 处理贸易伙伴间的商务活动, 使企业达到提升商业经营效率、降低成本、提高服务质量、增强对市场变化的快速反应能力、拓宽业务范围、提高竞争能力等一系列指标, 实现增加利润的目的。同时, 将极大地影响人们(消费者)的消费方式和生活方式。

电子商务的要素或特征可概括为 **2P + 3C**, 即五个要素:

(1) 以计算机网络, 特别是内联网、外联网和 **Internet** 为环境或平台 (Platform);

(2) 与贸易伙伴为协调和协作方式 (Collabration 和 Cooperation);

(3) 围绕贸易或商务主题 (Commerce);

(4) 对商务内容和信息做计算机化处理 (Content 和 Message);

(5) 实现利润——达到企业生存和发展的最终目标 (Profit)。

电子商务的概念早在**电子数据交换 (EDI)** 技术出现时就开始了。20 世纪 80 年代中期, 一些发达国家 EDI 的应用形成了一定规模, 引起了全球范围进行所谓“无纸贸易”的热潮。到了 90 年代, **Internet** 在全球迅猛发展, 企业通过计算机网络, 特别是 **Internet**, 进行的商务活动迅速增加, 成为数字化和网络经济时代解决企业经营管理、生存发展问题的一个重要途径。1997 年 11 月 6 日—7 日在法国巴

黎由“国际商会”召开的“世界电子商务会议”对电子商务的概念进行了探讨。1999 年 12 月 14 日, 在美国旧金山的 St. Francis 饭店, 公布了世界上第一个 **Internet** 商务标准。

电子商务的应用范围极为广泛, 因此有很多不同的分类方法, 其中最基本的方法是从电子商务的交易对象来分类: ①企业与企业之间的电子商务 (B2B) 供求企业之间以及协作企业之间利用网络交换信息、传递各种票据、支付货款, 从而使商务活动全过程实现电子化。②企业和消费者之间的电子商务 (B2C) 其典型应用是网上购物, 即电子化的销售。③企业和政府之间的电子商务 (B2G) 其电子商务活动可以覆盖企业、公司与政府组织间的各种事务, 如政府采购清单通过 **Internet** 发布, 企业、公司以电子化方式来完成对政府采购的响应。④消费者与政府之间的电子商务 (C2G) 政府可以把电子商务扩展到税款征收和社会医疗保险等方面, 如税务征管部门通过网络进行个人所得税及其他一些税务的申报、征缴等。⑤企业内部的电子商务 其电子商务活动以企业内联网作为一种安全、有效的工具, 用来自动处理商务操作及工作流程, 实现企业内部数据库信息共享, 并为企业内部的通信联系提供快捷的通道。企业内联网的商务应用可以增强企业商务活动的敏捷性, 使企业对市场的变化作出更加灵敏的反应, 为客户提供更全面、优质和高效的服务。

实现完善的、真正意义上的电子商务, 需要一个完整的电子商务技术体系作为基础, 包括: 数据通信技术、网络互联技术、信息与网络安全技术、电子数据交换 (EDI) 技术、数据库技术、电子支付技术、电子交易认证技术、标准化技术等。

参考文献

Schnneider G, Perry J. Electronic Commerce (2nd Edition). Thomson Learning, 2002

(史美林 孙志挥)

dianzi shangwu biao zhun

电子商务标准 (the standard for electronic commerce) 规范电子商务活动的国际标准。1999 年 12 月 14 日, 在美国旧金山的 St. Francis 饭店, 公布了世界上第一个 **Internet** 商务标准 (The Standard for Internet Commerce, Version 1.0—1999)。这一标准是由 Ziff-Davis 杂志牵头, 组织了 301 位世界著名的 **Internet** 和 IT 业巨头、相关记者、民间团体、学者等经过半年时间, 对 7 项、47 款标准进行了

两轮投票后才最终确定下来的。虽然这还只是 1.0 版,但它已经在相当程度上规范了利用 Internet 从事零售业的网上商店需要遵从的标准。虽然它完全是按照美国标准制定的,但显然对我国正在起步的电子商务事业有相当的参考价值。

在这一标准中首先定义了**电子商务**和**Internet 商务**概念。

Internet 商务是指利用 Internet (包括 WWW 万维网)进行的任何电子商务运作。

制订这个 Internet 标准的目的有 5 个:①增加消费者在 Internet 上进行交易的信心和满意程度;②建立消费者和销售商之间的信赖关系;③帮助销售商获得世界级的客户服务经验,加快发展步伐并降低成本;④支持和增强 Internet 商务的自我调节能力;⑤帮助销售商和消费者理解并处理快速增加的各种准则和符号。显然,这一标准既可以被销售商用于其 Internet 商务,并且向所有消费者和合作伙伴宣称自己符合这一标准,也可以被消费者用来检验销售商是否可以提供高质量的服务。同时还可以指导信息技术(IT)供应商、网站开发商、系统集成商等从事相关的业务。

整个标准分 7 项 47 款。每一款项都注明是“最低要求”,还是“最佳选择”。如果一个销售商宣称自己的网上商店符合这一标准,那它必须达到所有的最低标准。标准内容如下:

一、信息中心

(1) 必须建立一个信息中心,并且使消费者在网站上的任何地方都可以找到这个信息中心的链接。(最低要求)

(2) 销售商必须使用“Information”这个词作为该信息中心的标题。(最低要求)

二、需公布的内容

(3) 销售商必须在信息中心公布如下内容:销售商的法定名称以及业主、主要办公地点、与销售商联系的渠道(如电话或 E-mail);特殊业务的专业许可证。(最低要求)

(4) 必须在信息中心提供在广告中没有明确的客户支付方式或其他使用第三方产品或服务的资料。(最低要求)

(5) 在消费者被要求最终确认订单之前,销售商必须为消费者提供所有费用的清单,包括商品-服务的费用、运费、处理费以及税。(最低要求)、

(6) 信息中心必须提供质量保证的说明,包括担保的有效期限、适用的范围、不适用的范围、如何担

保等。(最低要求)

(7) 对每项产品或服务都必须提供有关售后服务的信息,包括服务范围、期限、如何进行等。(最低要求)

(8) 在信息中心中必须向客户说明适用哪一国家或地区的法律。(最低要求)

(9) 必须向客户公布可以选择的各种支付方式。(最低要求)

(10) 必须提供有关处理取消订单、退货、退款的原则,包括可以取消订单的有效期限、可以退货的产品、退货的条件、取消订单或者退货的费用、运费的支付方、消费者何时可以得到退款等。(最低要求)

(11) 必须公布销售商从消费者的信用卡上收款的规定。(最低要求)

三、产品与服务

(12) 如果销售商在销售或发货上对不同消费者(如特定的地区或年龄)有限制必须明确说明。(最低要求)

(13) 在消费者最终订货之前,销售商必须提供有关产品的供应情况,即货物发送或订单的处理估计所需要的时间。(最低要求)

(14) 销售商必须在两个工作日内通知期货订购者。(最低要求)

(15) 必须用明显的标记,如用颜色、图标等标识那些在网站上列出、但不能从网络上直接订购的商品。(最低要求)

(16) 对于订购了无现货的产品的消费者,应该在货物到达后通知他们。(最佳选择)

四、保密和安全

(17) 必须公布销售商的保密原则,至少包括:销售商将收集消费者的哪些资料,在何处收集;使用这些资料的目的;销售商是否会向第三方提供这些资料,如果提供,是在何种情况下;消费者资料是否是整个商务计划的一部分,如进行目标市场分析、建立各种促销方案等;消费者是否有可能限制使用私人资料,如何进行。(最低要求)

(18) 销售商必须在主页和信息中心提供标记为“Privacy”的保密原则链接。(最低要求)

(19) 消费者必须有能力选择是否使用销售商利用收集到的消费者资料主动发送的各种信息,并且在这些资料被开始收集时就可以进行这种选择。(最低要求)

(20) 消费者必须有能力选择是否同意将自己的私人信息提供给第三方,并且在这些资料被开始

收集时就可以进行这种选择。(最低要求)

(21) 如果有关交易的第三方(如购物车、支付网关)的保密原则与销售商的不同,销售商必须提供指向第三方保密原则的链接。(最低要求)

(22) 在整个交易过程中,销售商必须对所有消费者提供的信息进行加密传输。(最低要求)

(23) 销售商必须对销售商储存的消费者资料进行加密处理。(最低要求)

(24) 在信息中心,销售商必须为消费者提供哪些传输过程和资料是被保护的信息。(最低要求)

五、确认和通知

(25) 销售商必须在消费者订货后一个工作日内向消费者发出订单确认电子邮件。(最低要求)

(26) 销售商必须将总费用包括在订单确认通知中,或者明确告诉消费者从何处可以查找到总费用。(最低要求)

(27) 销售商应该在消费者订购的货物被发运或者服务被执行后一个工作日内通过电子邮件通知消费者。(最佳选择)

(28) 销售商必须将如下信息包含在发运通知中,或者明确告诉消费者从哪里可以得到这些消息:货物名称、总费用、货物从哪里以何种方式运出、估计的运输时间和如果有问题如何解决。(最低要求)

(29) 如果消费者选择的运输方式是可以进行货物在运输过程中的跟踪,销售商也应该为消费者提供这一方法。(最佳选择)

(30) 如果消费者选择的运输方式提供有关货物已经被收取和收取者姓名的资料,销售商也应该为消费者提供这一方法。(最佳选择)

(31) 如果销售商仅运出消费者订购的部分商品时,销售商应该通知消费者其他商品将在以后运出。(最佳选择)

(32) 如果客户取消订单或者退货,销售商必须在三个工作日中通知消费者已经收到取消订单或者退货。(最低要求)

六、帮助和客户服务

(33) 销售商必须为消费者提供通过电子邮件提问或投诉的渠道。(最低要求)

(34) 销售商必须在信息中心中提供获得客户服务条款的渠道。(最低要求)

(35) 销售商必须提供客户反馈和文本投诉的渠道。(最低要求)

(36) 销售商必须在收到问题或投诉 48 小时内

向消费者承认收到了问题或投诉。(最低要求)

(37) 如果投诉的是有关商品问题而且销售商自身不能解决,销售商必须向消费者提供与生产商联系的适当方法。(最低要求)

七、其他

(38) 销售商应该保证发运的每个包装都在运输机构进行了标准的防丢失、防盗和防损害保险。(最佳选择)

(39) 销售商必须按照可打印的格式向消费者提供订购货物的发票。(最低要求)

(40) 如果消费者选择的运输公司许可,销售商应该向消费者提供可以进行特别投递的能力。(最佳选择)

(41) 如果消费者以前已经提供过必要的信息,销售商应该为消费者提供“一键”购物的能力。(最佳选择)

(42) 电子商务模式语言(ECML)的支持,允许消费者在填写购物车表格时,避免重复性的输入。(最佳选择)

(43) 销售商应该提供实时处理订单和校验消费者信用卡的能力。(最佳选择)

(44) 销售商应该提供通过关键词对整个站点的信息和产品进行搜索的能力。(最佳选择)

(45) 销售商应该提供消费者可以通过万维网来检查订单状况的工具。(最佳选择)

(46) 销售商应该提供消费者可以检查以前订单的能力。(最佳选择)

(47) 销售商应该有一种系统化的方法来不断处理消费者的反馈和了解他们的满意程度。(最佳选择)

(史美林)

dianzi sheji zidonghua

电子设计自动化 (electronic design automation, EDA) 利用计算机来辅助设计电子器件、电路、装置和(或)系统的技术。现代计算机或微电子器件从总体方案的论证(系统级模拟)到指令序列的论证(行为功能级模拟)、逻辑设计(逻辑综合)、逻辑图检查(逻辑模拟)、布图(布局布线)和测试等全部采用**计算机辅助设计(CAD)**、**计算机辅助制造(CAM)**、**计算机辅助测试(CAT)**技术。长期以来,将计算机在超大规模集成电路(VLSI)或电子数字系统设计这一特定领域中的CAD/CAM/CAT统称为**设计自动化(DA)**,通常亦称为**电子设计自动化(EDA)**。应用EDA技术,可以显著减少从设计到制

造完成的成本和时间,而且还能解决手工无法解决的电子产品设计的复杂问题。因此,EDA的应用和发展,是现代计算机和超大规模集成电路(VLSI)发展的基石和巨大动力。没有先进的EDA系统,现代计算机与超大规模集成电路就难以发展。

电子设计自动化始于20世纪50年代中期。当时计算机使用的还是晶体管,为了满足军用计算机的合同期限,各主要制造商都开始用计算机辅助设计。1956年提出了第一个辅助系统,用于逻辑方程检查、电路扇入扇出核对、机架布线长度计算、接线表编排以及底板布线等工程设计。从此开创了电子CAD的新纪元,并在60年代获得了迅速的发展。进入70年代,CAD在计算机的工程和设计方面,如逻辑划分、布局和布线,获得了成功的应用,EDA系统开始形成商品。技术研究领域从解决工程设计的繁琐、重复、工作量大、强度大的问题,扩展到高层次的设计自动化问题,诸如硬件描述语言、逻辑综合、电路分析、逻辑图自动生成、自动布局、布线、集成电路版图设计、故障自动测试和诊断、设计规则、电气规则、布图规则、硅编译器等研究。80年代到90年代是EDA走向成熟发展的年代,其应用覆盖电子系统设计的各个层次,从概念设计、行为综合和模拟、逻辑设计、工程设计,直到热分析设计等。EDA产品从微机、工作站到巨型机EDA系统,为各类工程人员及厂家广泛应用。电子设计自动化的最终目的是使电子系统设计、制造、测试实现自动化。随着该项技术的应用和深入发展,不断提出新的课题。它的主要研究内容有:

(1) 超高速集成电路硬件描述语言(VHDL)是一种描述电子系统硬件行为与结构的硬件设计语言。应用该语言,能清晰地表达电子元件(门级、芯片级、插件级及系统级)的硬件设计,并能进行逻辑综合、模拟、测试、维护和修改设计(参见**硬件描述语言**)。

(2) 逻辑综合技术 根据给定的技术要求,使实现的电路系统逻辑优化。目的是减少器件和连线冗余、简化版图,从而减少基片面积、提高产品速度、降低成本(参见**数字系统逻辑综合技术**)。

(3) 模拟技术 借助计算机,在芯片或计算机数字系统制造之前,检测出逻辑设计的错误(包括逻辑的、定时的错误);或者对电路的直流特性、温度、时序、频率、灵敏度和噪声等进行分析,用来检查电路元件参数在设计时选择是否合理,并可进行破坏性模拟试验。前者称为逻辑模拟,后者称为电路

模拟或分析技术(参见**数字系统模拟技术**)。

(4) 布图技术 按照特定的设计规则,把设计好的电路图转变为掩模图或工艺图。主要内容包括版图布局、设计规则、布线、通孔最小化、版图验证、时延分析和优化设计(参见**布图技术**)。

(5) 可编程逻辑阵列设计 根据用户的功能要求,在由集成电路厂家生产而未经过编程连接的成品器件(称母片)上,由用户按一定的设计规则自行设计并实现母片的编程连接,制成所需要的专用集成电路。

(6) 测试诊断技术 也称故障测试诊断。测试指检测电路故障的存在;诊断则是确定故障在电路中的位置。研究的主要内容是测试矢量的生成和测试验证问题。随着超大规模集成电路(VLSI)复杂度的增加,常规的测试方法很难奏效。易测试性技术正日益受到重视。

(7) 热分析建模及模拟技术 指建立电子元件通电发热数学模型,应用模拟软件分析,显示电子设备装置的热分布状态,从而帮助设计师找出热点,以便改进布局设计。

EDA的应用领域主要有:①通用集成电路设计;②专用集成电路设计;③计算机设计;④电信设备设计;⑤其他复杂电子装置系统设计。

国内对EDA技术研究始于20世纪70年代中期,在两大领域展开研究。一是印制线路板计算机辅助设计和制造(PCB-CAD/CAM);二是集成电路CAD系统。经过20多年的研究、开发和攻关,我国已拥有自己的集成电路CAD系统,并在工厂投入实际使用;拥有自己版权的多种PCB-CAD/CAM系统,在电子行业获得广泛应用。这对普及EDA技术起着促进作用。这些系统正朝着商品化、集成化方向发展。

国外EDA系统20世纪70年代初开始商品化,目前已普遍使用。80年代解决集成化、同族EDA工具之间的数据共享问题。80年代后期开始EDA框架技术研究,解决异族电子CAD工具的集成技术问题。主要包括数据模型、设计过程控制、工具封装与自动激活、EDA框架用户接口等。90年代中期开始高级逻辑综合行为模拟技术和片上系统设计(SOC)技术研究,近几年又开始集成开发平台(IDE)研究。

参考文献

1. 刘明业. 数字系统设计自动化. 北京: 电子工业出版社, 1994
2. 刘明业等. 专用集成电路高级综合理论. 北

京:北京理工大学出版社,1998

(潘雪增)

dianzi shuju jiaohuan

电子数据交换 (electronic data interchange,

EDI) 发票、订单等结构化数据,在计算机间按标准化格式进行交换和自动处理的技术。在某些情况下,EDI也可以是计算机至某些物理复制系统的结构化数据的传送,例如由计算机传送到物理投递系统或传真机。EDI的实质是通过用户原始数据的标准化表示,实现数据经通信网络在各用户所拥有的计算机应用系统之间的交换和自动处理,达到迅速和可靠的目的。采用EDI技术,将取消传统的纸张文件,因而在贸易方面使用的EDI又称为“无纸贸易”。

EDI工作流程分为3个部分:①文件结构化和标准化处理,即用户将原始的数据文件经计算机处理,形成具有标准格式的EDI数据文件;②传输和交换,即将标准EDI数据,经过EDI数据通信和交换网,传送到对方用户的计算机系统或物理复制系统;③文件接收和自动处理,即对方用户计算机收到发来的标准EDI数据后,立即按照特定的程序自动进行处理,将人工干预减少到最低程度。EDI工作流程图如图1所示。

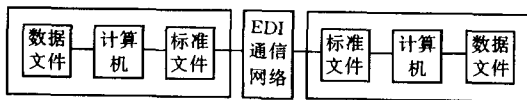


图1 EDI 工作流程图

由此可见,EDI技术主要包括EDI标准化、软件系统、通信支撑和系统实现等,可概括为EDI的三个要素:标准化和结构化的数据;计算机网络的传输;整个过程的完全自动处理。

标准 实现数据文件的互换和自动处理,文件结构、格式、语法规则等方面的标准化是实现EDI的关键。目前国外已经形成众多的EDI行业标准,其中具有代表性的有两大标准体系,一个是广泛用于北美的美国国家标准ANSI X.12,已有140多个不同的变体,大约在8000个组织中使用;另一个是联合国制定的UN/EDIFACT。EDIFACT已被国际标准化组织(ISO)注册为国际标准,成为EDI标准发展的主流,目前已有EDI数据类型35种(计划70种左右)。一些EDI团体组织正纷纷向UN/EDIFACT标准靠拢,我国已明确表示采用该标准。

标准EDI数据是由数据元、数据段、交易集和功能组构成的。若干数据元构成一个数据段,若干数据段构成一个交易集,若干交易集构成一个功能组。

软件组成 一个EDI软件系统可以有如下几个部分组成:①人机接口模块,便于用户进行系统管理和操作。②内部接口模块,指EDI系统和其他信息系统与数据库的接口。③消息生成和处理模块接收来自用户接口模块的命令和信息,生成EDI数据,经EDI网络转发给其他EDI用户,或者自动处理其他EDI系统发来的EDI消息。④格式转换模块将待发的应用数据转换成标准EDI消息格式,或者将接收的EDI消息转换成应用数据格式。格式转换包括语法上的压缩和嵌套、代码替换、添加EDI语法控制字符等,同时还要进行语法检查。⑤通信模块是EDI系统与EDI通信网络的接口,其基本功能有执行呼叫、自动重复、合法性和完整性检查、出错报警、自动应答、通信记录、消息组装和拆卸等。

系统实现 EDI交换原则上可以采用任意的通信网络,例如电话网、公共数据网、专用数据网或增值网。国际电报电话咨询委员会(CCITT)X.400系列建议的消息处理系统(MHS)是实现EDI交换的良好传输系统,它提供消息传输服务和用户间消息通信服务。为了适应EDI应用的需要,CCITT补充了X.435建议,通过定义EDI用户(计算机应用进程)访问消息传输系统的接口和支持EDI用户代理(EDIUA)之间交互的通信协议Pedi,来构造EDI消息通信系统(EDIMS),借助于消息传输服务,实现EDI用户之间的数据交换。

EDIMS的工作方式是,EDI用户利用EDI用户代理组织EDI消息,装入“信封”,提交给消息传输系统。信封信息与传输相关,由消息传输系统处理。EDI消息包括信首和信体两部分,信首参数标识特定的EDI消息,信体主要是EDI用户交换的信息。当EDI消息被投递到收方的EDI用户代理时,收方的EDI用户代理可以产生EDI回执(EDIN)返回给发方。收方EDI用户根据应用要求,自动处理收到的数据或生成新的数据,作为新的EDI消息,继续进行EDI交换。

通信平台 实现EDI有两个层次的工作,一个是通信方面,包括传输和信箱;另一个是EDI业务管理方面,包括EDI软件标准、入网管理、用户终端软件开发和数据文件传递的法律效力等。后者涉及本地EDI系统与其他远地EDI系统乃至整个EDI

网络系统的关系。因此,EDI 通信应能够满足 EDI 业务管理要求,具有良好的 EDI 网络适应能力,即 EDI 的通信平台必须具有下列几方面的开放性:

①EDI 入网方式的开放性,包括 EDI 信箱可与多种用户计算机系统相连和支持多种通信协议;②EDI 软件平台的开放性,即 EDI 标准的开放性和应用系统的开放性;③ EDI 网络组织和业务组织的开放性。

社会影响 EDI 已广泛应用于各个领域,例如商业贸易、海关业务、金融交易、办公自动化、文教卫生、工业制造、纺织及服务业。EDI 是现代电子技术与管理制结合度的产物,是提高工作效率和降低业务成本的有效手段,是现代社发展的必然趋势。

EDI 作为一种新型的业务处理手段,在迅速、安全、可靠和低成本等方面具有传统的业务方式所无法比拟的优越性,将对传统的业务方式和行政管理方式产生重大影响。例如一些公章和纸面文件将被取消,原有的管理体制将根本改变。EDI 是计算机和通信技术应用日益深入发展的必然产物,是一场结构性产业革命。EDI 的推广和实施直接涉及诸如电子签名、法律证据和仲裁等法律问题,必须制定相关的法律、法规和 EDI 立法。

EDI II EDI II 称为**企业数据集成**,由 EDI 演变而来。EDI II 是数据文件按标准化格式在应用到应用之间的交换,形成业务处理过程的重新设计和集成。EDI 保证数据的快速、准确和高效交换,而 EDI II 强调业务伙伴间的业务处理过程,是基本 EDI 环境中业务过程的进一步集成。EDI II 面向应用到应用,重新考虑有关业务处理过程中原有的信息流、任务和责任。EDI II 所产生的变化不仅影响到数据库和应用逻辑,而且使传统的业务过程发生根本变化。

EDI II 不只是几个业务伙伴日常操作管理应用系统的集成,更需要不同业务间操作实践、目标、政策、作用、责任和文化的集成。这种形式的集成必须以各业务伙伴认可的业务条件为前提,例如及时生产(JIT)和快速响应(QR)。

参考文献

1. Kimberley P. Electronic Data Interchange. New York: McGraw-Hill Inc., 1991

2. Barber N. EDI II and Trading Partners. Interchange, 1994, 2: 7~9

(罗军舟)

dianzi yinqian chuli

电子印前处理 (electronic pre-press processing) 以计算机软、硬件为核心,完成印刷前排版、制版、打样等工序的过程和技术。实现上述技术的电子印前系统一般包括计算机、图像扫描仪、光栅图像处理器、激光照排机,以及用于打样的彩色或黑白打印机和各种文字图片排版、制版软件。电子印前处理综合应用了**数字图像处理技术**、**计算机图形学**、**计算机辅助设计**、**人机交互技术**、**色度学**、**印刷色彩学**、**彩色管理技术**、**电子分色技术**、**电子激光成像技术**等。

印刷过程一般分为印前处理、印刷和印后加工 3 个阶段。电子印前处理的功能是利用电子技术和相关设备(特别是计算机和软件)完成排版、制版、打样等任务。它对印刷行业的技术改造与技术进步具有重要的意义。从文字印刷角度讲,它使人们摆脱了传统的铅字排版工艺;从彩色印刷角度讲,它使人们摆脱了手工修版、手工拼版等繁杂的传统工艺。

1980 年以前,计算机的主要输出设备是打印机和绘图仪。那时就有人将打印机和绘图仪的输出结果翻拍成胶片,然后制版印刷。这样的印刷品,一般质量较差,且内容仅限于文字和图形,但免去了低效的手工铅字排版。

随着与计算机联机使用的数控光电图像扫描设备,数控光学胶片记录设备,特别是数控激光胶片记录设备——激光照排机的发展与完善,在 20 世纪 80 年代初,出现了能进行文字排版、处理黑白图片的早期电子印前系统。

20 世纪 80 年代是电子印前系统研究开发最活跃的 10 年,也是电子印前系统走向成熟的 10 年。高分辨率、高精度的**激光照排机**,高分辨率的黑白和彩色扫描仪,处理文字(包括汉字)、图片挂网和图形的光栅图像处理器,文字排版软件,图像编辑拼版软件,可用于打样的各种黑白和彩色打印机,先后被研制开发出来。用于电子印前系统的**页面描述语言**也逐步成熟并形成国际标准。与此同时,彩色印刷中的关键设备——电子分色机,完成了从模拟电子分色机到全数字电子分色机的演变,并研制出了电子分色机与计算机的接口设备及相应的拼版、修版软件。

20 世纪 90 年代起,随着计算机科学技术的进步,电子印前技术得到了飞速发展。真正具有实用意义的彩色电子印前系统问世,电子印前系统在印

刷业的地位得到确立,电子印前系统也进入了一个新的发展时期。

电子印前处理系统有两种类型:精密照排系统和桌面出版系统(DTP)。精密照排系统由计算机、栅格图像处理器、激光照排机以及其他相关设备组成。它配置了文字处理、排版软件、图像处理软件以及输出软件,用于书刊、报纸等专业出版印刷领域的各种出版物的输入、编辑、排版和打样。桌面出版系统以个人计算机为基础,配置相关的软件和硬件设备,它能采集、处理图像和文字信息,并将处理好的图像与文字一起组入版面内,然后整版输出。DTP系统具有“所见即所得”的特点,操作简便、输出版面质量高,在办公领域广为使用。随着个人计算机性能的不断提,软件功能的日益完善,彩色图像输入输出设备成本的不断下降,DTP系统的性能已经达到专业应用的要求,成为当前电子印前系统中的主流。

电子印前处理系统的组成如图1所示。其中图、文输入设备用来输入排版用的文字和图形素材。图像可通过扫描仪、数字照相机、摄像机等变成数据文件储存在计算机内。而图形则可使用绘图软件生成。文字可以用键盘录入,也可通过文字识别或者言语识别技术直接获取。

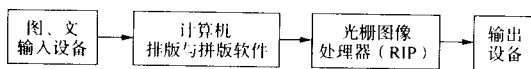


图1 电子印前处理系统的组成

排版与拼版功能是在计算机上运行专门的排版与拼版软件来完成的。排版软件把输入的文字、图形、图像等版面元素按照需要放置到页面上的指定位置,并形成艺术效果。处理后的结果通常是使用页面描述语言表示的文件。

光栅图像处理器也称栅格图像处理器(RIP),它将排版结果转换成对应于物理输出设备的光栅点阵。这一过程对输出的质量和速度有着决定性的影响。光栅图像处理器是一种大规模集成电路制成的专用处理器,在低成本的DTP系统中它也可以通过软件来实现。

输出设备包括各种高精度打印机,如激光打印机、喷墨打印机、彩色电子打样机、激光照排机等。其中通过激光照排机输出的软片可作为印刷制版之用。

对电子印前系统进行研究的内容可分为电子印前系统的专用计算机硬件和专用计算机软件两个方

面。专用硬件的主要研究内容有:①光栅图像处理器;②加速图像处理的高速专用芯片和专用硬件;③图像扫描仪;④激光胶片记录仪,即照排机;⑤可用于打样的各类打印机;⑥与电子分色机接口的专用硬件。专用软件的主要研究内容有:①彩色管理技术及其软件;②分色技术及其软件;③挂网技术及其软件;④拼版、修版软件;⑤页面描述语言及其解释软件;⑥文字排版软件等。

我国电子印前系统的研究与开发工作是在1974年8月国务院批准《关于研制汉字信息处理系统工程的请示报告》后开始的,北京大学、山东潍坊计算机公司等单位开始研制精密照排系统,经过多年努力解决了一系列关键技术问题,如提出了一种数据压缩倍数很高又能确保字形质量(不失真)的汉字字形描述方法;开发了专用硬件实现汉字字形的复原及放大缩小;采用最新技术研制激光照排机;排版软件采用自动生成复杂版式的方法;系统采用统一的页面描述语言等。成功地开发出了华光及方正系列精密照排系统。

经过不断的积累和创新,目前,我国以北大方正电子有限公司为代表,在电子印前系统的研究与开发的许多方面已接近国际先进水平,有些领域处于国际先进水平。

参考文献

北大方正电子有限公司网站: <http://www.founder.com.cn/> 2003 (陈晓鸣 张福炎)

dianzi youjian

电子邮件(electronic mail, E-mail) 在发送者和指定的接收者之间,利用计算机网络进行的文本、数据、图像或语音(言语)等信息的非交互式通信。在计算机网络众多的服务中,电子邮件是最广泛、最有发展前途的服务之一。据统计,在Internet提供的服务中1/3以上是电子邮件。电子邮件系统既是一种通用的网络应用,也是一种为其他应用所使用的基础设施。

在电子邮件发送前,每个用户必须有一个电子邮箱来存放邮件。每个邮箱有一个惟一的邮件地址,它分为两部分:第一部分标识用户的邮箱,第二部分标识邮箱所在的计算机。一种广泛使用的格式是 mailbox@computer,这里 mailbox 是一个指明用户邮箱的字符串,而 computer 是一个指明邮箱所在的计算机的字符串。

简单邮件传送协议(SMTP)是TCP/IP协议集

中用于主机之间相互传送邮件的标准协议。SMTP 采用了 RFC 822 定义的电子邮件发送的文本报文格式,包括信封和报文内容。信封包含的是各种用于完成邮件传输所必要的信息。传统的 SMTP 仅局限于传送简单的文本报文。

通用 Internet 邮件扩展协议 (MIME) 是 RFC 822 框架结构的扩展,可传输多媒体信息、包含各国语言文字的文本数据、可执行文件或其他二进制文件等。

(胡道元)

dianzi zhengwu

电子政务 (electronic government) 政府机构运用现代网络通信与计算机技术,将其管理和 Service 职能通过精简、优化、整合、重组后在互联网上实现的过程与技术。电子政务可以打破时间、空间以及条块分割的制约,加强对政府业务的有效监管,提高政府的运作效率,并为社会公众提供高效、优质、廉洁的一体化管理和服务。

电子政务的特征主要有:①它是一个在政府业务流程优化的基础上,包含通信网络、计算机硬件和软件、政府工作人员、社会公众和企业在内的人机结合的应用系统和社会系统工程。②安全是其最为重要的基础。电子政务安全体系的内容不仅包括通过技术手段保证网络安全和信息安全,还包括安全管理制度的建设和对政府工作人员安全意识的增强。③以互联网为基础运行环境。互联网本身所具有的开放性、全球性、低成本、高效率是电子政务的内在特征(参见 Internet)。④其建设和应用效果依赖于政府各部门业务信息化的程度。⑤能够满足新经济条件下公众和企业对政府的要求,使政府由一个多层次、多部门的“管理型”政府向一个智能化、高效、公开、透明、低成本运行的“管理服务型”政府转变。⑥以政府机构(包括政府工作人员)、企业和社会公众(居民)为主体展开政府业务活动,即包括政府与政府之间的互动,政府与企业单位的互动以及政府与居民的互动。⑦电子政务是一项政府管理创新和实践的过程,在世界各国均处于不断迅速发展的过程中。

电子政务概念的内涵,经历了一个发展变化的过程。20 世纪 80 年代前后,首先提出了**办公自动化**的概念。其核心是用计算机技术处理办公室的内部业务,例如文件资料的制作、传送和储存等。80 年代以后,随着管理信息系统的出现,需要对传统的政府管理和公共服务进行改造,提出了运用信息加

工和信息处理技术改善政府的决策和满足管理者的需求,即政府信息化。90 年代后,随着互联网技术的发展及在政府公共管理中的应用,引出了电子政务的概念。其时电子政务的含义是指在政府内部办公自动化(包括公文处理、电子邮件、签报管理、会议会务管理、案卷管理、日程安排、公共信息服务、电子论坛等)基础上,利用计算机技术、通信技术和网络技术,建立网络化的政府信息系统,并通过不同的信息服务设施和网络,计算机及电话等工具,为企业、社会乃至公民个人提供政府信息和其他公共服务,改变政府管理的方式。可见电子政务与政府信息化、政府办公自动化有着密切的联系。电子政务是实现政府信息化的一种主要手段。电子政务是政府全局性、全过程、综合业务的自动化,而办公自动化则侧重于政府内部事务处理的自动化。

电子政务建设是一项系统工程,涉及国家不同级别的政府权力机构、政府工作人员、社会公众等不同实体,各个实体对应不同级别要求的安全保密、公众服务和决策支持。电子政务的总体体系架构具有以下特点:①网络基础设施环境要求高于一般信息系统,必须满足内外网络结构体系;②统一的安全支撑体系贯穿于从网络层、系统层乃至应用层的电子政务的所有环节;③标准化支撑体系是电子政务建设的根本原则之一;④必须以共享、一致的政务信息资源数据库作为电子政务信息系统的集成和运行的基础。

电子政务应用组件平台可基于分布式多层构架和组件技术构建,主要包括以下方面的组件:①工作流管理系统;②统计报表系统;③电子表单系统;④内容管理系统;⑤全文检索系统;⑥政府信息综合交换平台;⑦统一认证和授权系统;⑧统一用户和部门管理;⑨安全加密系统;⑩电子邮件系统;⑪统一消息平台。

电子政务业务应用系统一般包括:①**党政管理信息系统**;②内网运行的政府办公自动化系统;③专网上运行的行业管理业务应用系统;④外网上运行的公共监管和服务应用系统;⑤**国家级政务信息系统**。

参考文献

汪玉凯,赵国俊. 电子政务基础. 北京:中软电子出版社,2002

(孙志辉)

dingli jiqi zhengming

定理机器证明 (mechanical theorem pro-

ving) 对数学定理,用计算机自动进行的推理和证明。又称自动定理证明(ATP)。让机器去证明数学定理的想法,在17世纪 G. W. Leibniz 创立数理逻辑时就产生了,但这一想法的真正实现,是在20世纪40年代计算机诞生以后。

从1956年 A. Newell, J. C. Shaw, H. A. Simon 发表他们的著名论文“逻辑理论机”算起,自动定理证明这个研究领域已经有近四十年的历史了。逻辑理论机是机械地模仿人类在证明命题逻辑定理时所用的推导过程。1959年 H. Gelemter 等人做出了几何定理证明机(GTM)。GTM 能够证明直线图形中的大部分高中的考试题,而且运行时间也常常与高中学生做题的时间相当。1959年 P. C. Gilmore 设计了关于谓词演算的机械证明程序。在1958年到1960年间王浩研究出关于命题演算和谓词演算的机械证明程序。1965年 J. A. Robinson 提出了归结原理。归结原理实质上是一条简洁的推理规则。使用这一条规则,对一阶逻辑中的任一个恒真公式,都将是可证的。L. Wos 是最成功地实现归结系统的研究者之一,为此,他获得首次 ATP 奖。1972年左右,以 L. Wos 为首的集体建立了一个以归结方法为主的自动推理系统 AURA,这个系统对于解有限数学,线路设计,程序正确性验证及形式逻辑等领域中的疑难问题,提供了帮助。归结方法是 ATP 领域中第一类方法——逻辑方法的代表。

ATP 领域中第二类方法——类人方法,有时也称为自然演绎或自然推导法。A. Newell 等人的逻辑理论机就使用了类人方法。1966年美国麻省理工学院的 L. Norton 建造了一个系统 ADEPT,该系统是一个关于群论的启发式定理证明系统,它使用了很多有效的启发式规则。这个系统的能力界限是能够证明:若 $x^2 = e$,则群是交换群。

ATP 领域中的第三类方法是所谓判定过程。判定过程是指:判断一个理论中的某个公式的有效性。1978年,中国的吴文俊关于平面几何和微分几何的定理机器证明方法(亦即吴方法)被认为是 ATP 领域中判定方法的一个最好的结果。使用这个方法,在计算机上证明和发现了平面几何中一些有趣的定理。

下面用简单而直观的例子说明 ATP 领域中这三类方法。我们知道一个定理:若 A 成立,则 B 成立;还知道一个事实: A 成立。欲证明一个结论: B 也成立。我们想做的事,在数理逻辑中,相当于去证明公式:

$$(A \wedge (A \rightarrow B)) \rightarrow B \quad (1)$$

是一个定理,亦即证明公式(1)是恒真的。

第一种方法:归结方法是反驳方法,要证明公式(1)是恒真的,相当于证明公式(1)的否定是恒假的,亦即证明

$$A \wedge (A \rightarrow B) \wedge \sim B \quad (2)$$

是恒假的($\sim B$ 表示 B 的否定)。 A 和 $(A \rightarrow B)$ 归结得 B , B 和 $\sim B$ 归结得恒假公式(即空子句)。因此,公式(2)是恒假的,所以公式(1)是恒真的,定理得证。

第二种方法:类人方法就是模仿人在证明定理时的思维过程,将原目标不断分解成易证的子目标,直到子目标都是 $(P \rightarrow P)$ 的形式。为了证明公式(1),只要能证明 $(A \rightarrow B)$ 或者 $((A \rightarrow B) \rightarrow B)$ 中的一个即可(这称为自然演绎法中的“或分支规则”)。 $(A \rightarrow B)$ 无法证明,考虑

$$((A \rightarrow B) \rightarrow B) \quad (3)$$

只要能证明 $(B \rightarrow B)$ 和 A 即可(这称为“反向链规则”)。而 $(B \rightarrow B)$ 可证,下面证明 A :使用定理(1)的前提,即是要证明:

$$(A \wedge (A \rightarrow B)) \rightarrow A \quad (4)$$

对公式(4)再使用“或分支规则”,只要能证明 $(A \rightarrow A)$ 或者 $((A \rightarrow B) \rightarrow A)$ 中的一个即可,而 $(A \rightarrow A)$ 可证,故式(4)可证,因此 A 可证,故式(3)可证,所以定理(1)可证。

第三种方法:判定方法不是直接去证明定理,而是去判定这个定理是否正确,对定理使用真值表法即可判定它是成立的。

上述三种方法较详细的论述可分别参见归结方法,自然演绎法,吴方法。

定理机器证明面对的是数学领域,这就决定了定理机器证明必然要面临巨大的困难。因此,从80年代开始,很多研究定理机器证明的学者,转向了使用简单推理就能获得成效的领域,从而诞生了人工智能中一个极为活跃的研究领域——自动推理。自动定理证明和自动推理在人工智能领域中越来越显出其重要性。

参考文献

1. 吴文俊. 几何定理机器证明的基本原理. 北京: 科学出版社, 1984
2. 刘叙华, 姜云飞. 定理机器证明. 北京: 科学出版社, 1987
3. Chang C L, Lee R C T. Symbolic Logic and Mechanical Theorem Proving. New York: Academic

Press, 1973

(刘叙华)

dingxing tuili

定性推理 (qualitative reasoning) 从物理系统的结构描述出发,导出行为描述和功能描述,预测物理系统的行为,并给出因果关系的解释的一种非定量推理方法。

量的定性描述是有明确结构的一种非定量的、非精确的表示方法,如物理系统变化趋势的定性描述常是有效的。定性推理是源于物理现象的一种常识推理,1977年 Reiger 发表了第一篇定性推理的论文,1984年《Artificial Intelligence》杂志出版了定性推理专辑,刊登了 D. J. Kleer, K. D. Forbus 和 B. Kuipers 等人关于定性推理的奠基性文章,标志着定性推理开始走向成熟。随后,定性推理得到人工智能界的普遍关注,得以发展。

传统的定量方法是精确描述物理量间的关系,但不具备推理能力,物理量之间的因果关系湮没在精确的数值中。而定性方法具备推理能力,能表达物理系统的因果关系,通过局部传播能在较高层次上给出系统的宏观描述。

结构描述和行为描述可理解为定量方程及其解的抽象,功能描述是对实际物理系统行为表现的一种理解。定性推理的论域是离散变化的符号集,最简单的定性论域是 $\{+, 0, -\}$,相当于把实数轴离散化为 $\{(-\infty, 0), [0, 0], (0, \text{eps}, +\infty)\}$,开域内的值表现了定性一致的行为性质,而边界值反映的则是转化点。

定性推理的基本方法如下:

Envision 方法 系统的结构用一个定性方程组来描述,系统的论域采用符号集 $\{+, 0, -, ?\}$ 。“?”的引入是为了保证运算的代数封闭性,表示不确定。系统的所有变量的一组取值构成一个状态,由初状态出发而得到的所有状态和状态转移构成了可能的展望空间。这种随时间变化的定性状态序列构成了系统的定性行为。

QPT 方法 系统用个体视图和进程来描述,个体是实际物理系统中存在的物理实体。个体视图是对个体及个体组合的抽象和描述。进程描述实际物理系统的变化。从实际物理系统的结构描述获得系统的行为描述就是推导实际物理系统视图结构和进程结构的过程。

QSIM 方法 系统的论域是被物理量的界标离散化的实数轴,结构描述由系统物理量之间的约束

组成。约束是物理量之间的定性关系,本质上是一组定性方程。推理过程是从系统的初状态产生所有可能的定性后继状态,根据约束关系从中选出各种相容的定性状态。重复这一过程,可得到实际物理系统的定性状态变化过程。

因果分析法 定性推理方法倾向于模拟人的思维方式,在对物理系统的行为求解的同时,要对行为做出解释,包括基于约束的因果分析、因果顺序分析和约束网络中的因果分析等。

定性定量相结合方法 是在定性的基础上引入定量信息,二者结合进行推理。

定性空间推理方法 涉及到刚体运动的几何性质,典型的如机器人运动的规划、导航。

定性推理有广泛的应用前景,如在诊断、设计等方面。

参考文献

Barr A, Cohen P R, Feigenbaum E A. The Handbook of Artificial Intelligence. Vol. IV. Addison-Wesley Publishing Company, 1989 (石纯一)

donghua houqi chuli

动画后期处理 (animation post-processing)

动画作品制作后期所采用的以非线性编辑、合成为主的过程和技术。它包括剪辑、叠化、抠像、合成、形态变换、特殊光效、配音、动画实时播放等。后期处理在计算机动画中占据非常重要的位置,是制作一个完整动画所必不可少的步骤。由于是全数码合成,避免了传统模拟编辑系统中因多次操作而引起的图像损失。已有许多应用于动画后期处理的优秀合成软件。在这些软件中,大量采用了图像处理和计算机视觉技术,可方便地进行特技效果的处理和运动跟踪、深度合成等复杂操作。

为了把计算机动画制作的虚拟场景与真实场景天衣无缝地合成在一起,需要考虑以下因素。①虚拟场景与真实场景的透视关系应一致。也就是说,虚拟场景的摄像机参数应与真实场景的摄像机参数完全相同。获取摄像机参数的一种方法是在拍摄时采用专用设备直接记录;另一种方法是在真实场景中放置标记,然后采用计算机视觉中的摄像机标定方法来反求摄像机参数。②虚拟场景的光照情况应与真实场景一致。例如,真实场景中的太阳,同样应照在虚拟场景上,使得虚拟物体呈现出落日的余晖。③虚拟场景与真实场景的相互作用。例如,虚拟恐龙踩在真实草地上引起草地的变化,真实场景

的风吹在虚拟狐猴身上引起其毛发的运动,虚拟场景投射在真实场景中的阴影等。④虚拟场景与真实场景的遮挡关系。虚拟场景既可能位于真实场景的前面,也可能位于其后面,也可能相互遮挡。因此,必须采用某些方法来确定场景相互之间的前后遮挡关系,如通过用户交互指定、给出场景的深度值等。

参考文献

1. 鲍虎军,金小刚,彭群生等. 计算机动画的算法基础. 杭州:浙江大学出版社,2000
2. Kelly D. Digital Compositing in Depth. Arizona: the Coriolis Group, 2000 (金小刚)

donghua miaoshu yuyan

动画描述语言 (animation script language)

一种用于描述动画的脚本语言。它使计算机动画系统更易为一般艺术工作者所接受,帮助他们创造出更多更动人的特殊动画效果。动画语言也经常与三维建模语言相结合,使得描述一个对象及其运动的过程可以同时完成。动画语言种类很多,主要可分为3类:线性表标记语言、扩展了动画描述功能的通用语言和图形语言。

在线性表标记语言中,对动画中每个事件的描述都由起始帧与结束帧的编号以及相关动作组成,并通过将这些事件组织成一个线性表来描述整个动画过程。这类动画语言在具体实现时往往有所扩展,例如 Scea 除支持线性表结构外,还提供了对象组和层次对象结构,以及类似高级语言的程序控制结构等。

另一种动画语言的设计方法是在现有的通用程序设计语言中嵌入动画描述,例如 ASAS 就是建立在 LISP 语言基础之上的一种动画描述语言。这类语言比线性表标记语言具有更强的表达能力,但要求使用者有一定的程序设计基础。

图形语言是一种更为可视化的动画语言,可以帮助使用者表达、编辑并理解动画中同步发生的变化。在这类语言中,标记的概念不再是文字描述,而是可视的图表,系统可以以图示的方式表现动作。

参考文献

1. 齐东旭,马华东,黄心渊等. 计算机动画原理与应用. 北京:科学出版社,1998
2. 唐泽圣,周嘉玉,李新友. 计算机图形学基础. 北京:清华大学出版社,1995 (金小刚)

dongtai sui ji cun qu cun chu qi xin pian

动态随机存取存储器芯片 (dynamic random access memory chip)

一种可以随机存取,但必须周期性地对其存储数据进行动态刷新,以防数据消失的一种存储器芯片。动态随机存取存储器 (DRAM) 芯片的存储单元将数字信息以电荷形式存于电容上,由于漏电,电容电荷会缓慢消失,导致读出数据错误。因此必须周期性地充电刷新,以保证所存数据不致丢失。DRAM 芯片所存数据将因断电而消失,属易失性存储器。

分类 通用 DRAM 芯片按接口可分为非同步 DRAM 和同步 DRAM (SDRAM) 芯片。作为基础的非同步 DRAM 芯片有快速页面模式 (FPM) 和数据扩展输出模式 (EDO) 两种工作方式。SDRAM 芯片的内核仍是非同步 DRAM 芯片的内核,但增加了同步时钟输入,并将地址、控制、数据输入输出的对外接口改为与时钟同步。按数据输入输出的速率,SDRAM 芯片又分为 SDR (单数据速率)、DDR (双数据速率) 和 DDR2 (4 倍数据速率) SDRAM 芯片以及 Rambus DRAM (RDRAM) 芯片。通用 SDRAM 芯片的封装、引脚排列和定义、内部构成有 JEDEC 标准,以保证通用性。

存储单元 DRAM 芯片的存储单元电路种类很多。由于单管单元所需器件少,对于相同的芯片面积,用单管单元的芯片可容纳更多的存储单元,因此单管单元的应用最为广泛。

图 1 为 DRAM 芯片的单管存储单元示意图。每个存储单元由一个 MOS 管和一个存储电容组成。MOS 管用于地址选择,其栅极接行 (字) 选择线,源极接列 (位) 数据读出放大和写入驱动线,漏极接存储电容。在集成电路制造过程中,存储电容与栅极同时制成,以减小面积。用电容的存储电荷表示存储的数据。若电容充电,则电容上电压高,表示 1; 电容未充电,则电容上电压低表示 0。

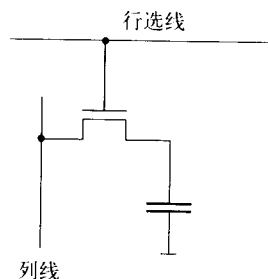


图 1 DRAM 芯片的单管存储单元

容量 DRAM 芯片的存储容量为 $2^N \times M$ 位,即字数为 2^N ,字长为 M 位, M 表示输入输出数据的位数。这样就确定了所访问存储单元的有效地址长度为 N 位。随着 DRAM 芯片容量的增大,芯片内部结构分为 $2^{N_B}(N_B=1,2,3)$ 个独立的存储体,每个体有各自的行译码和驱动及列译码和读出放大、写入驱动。存储体的地址 BA 为 N_B 位。每个体的存储单元排成 2^{N_R} 行 $\times 2^{N_L}$ 列阵列。 N_R 和 N_L 分别为行地址和列地址的位数, $N_R > N_L$ 。 $N = N_R + N_L + N_B$ 。数据位数 M 通常为 4, 8, 16 或 32。

为了降低成本,减少地址输入管脚的数量,

DRAM 芯片均采用分时地址方式,即行地址和列地址共用地址输入引脚,按时序先输入行地址,后输入列地址。分时地址输入和相应的时序控制信号由专门的接口控制电路完成。

组成 DRAM 芯片由存储单元阵列、行地址缓冲、列地址缓冲以及译码驱动电路、读出放大和写入驱动电路、数据输入输出缓冲、刷新地址计数与控制以及时序控制电路等组成。图 2 为 $4 \times 8 M \times 8 b$ SDRAM 芯片的电路框图(N_R 为 13, N_L 为 10, N_B 为 2, M 为 8, 页面大小为 1 024)。

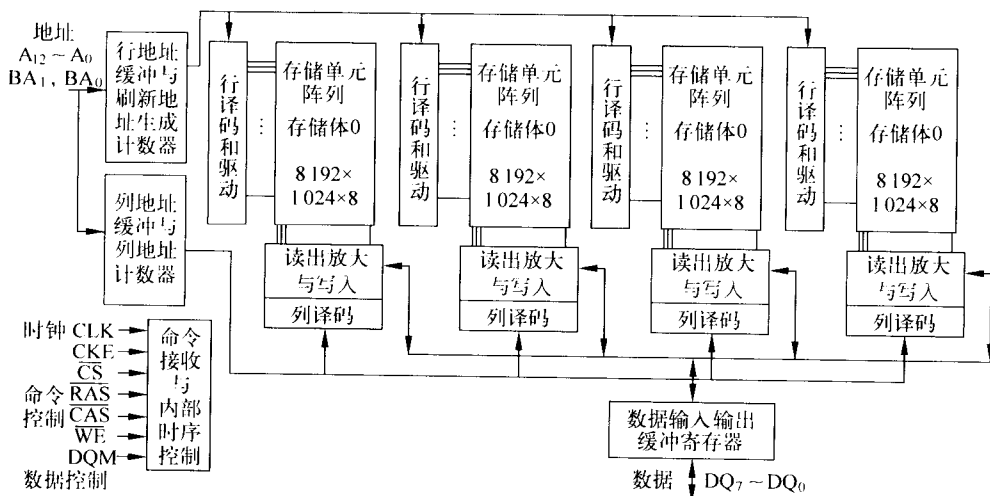


图 2 256Mb(4x8Mx8b)SDRAM 芯片框图

非同步 DRAM 芯片无时钟信号输入,以行选输入 \overline{RAS} 和列选输入 \overline{CAS} 为基本时序控制。当 \overline{RAS} 变为低电平时,下降边将行地址打入行地址缓冲。经过译码后,选中行线上所有存储单元的数据均输出到列数据寄存器。然后,当 \overline{CAS} 变为低电平时,其下降边将列地址打入列地址缓冲。经列地址译码,将选中的列数据寄存器中的数据送到输出数据缓冲。由于选中行线的所有存储单元的数据均输出到列数据寄存器,因此 DRAM 可以选择快速工作方式,或称为页面工作方式。即在 \overline{RAS} 变为低后, \overline{CAS} 连续有负脉冲,访问已选中行的不同列的存储单元。页面的大小为 2^{N_L} 。DRAM 芯片常用的数据输出方式有快速页面(FPM)与数据扩展输出(EDO)这两种快速工作方式。

SDRAM 芯片的内核仍是非同步 DRAM,但输入地址($A_{12} \sim A_0$, BA_0 和 BA_1)、命令控制(片选 \overline{CS} 、行

选 \overline{RAS} 、列选 \overline{CAS} 和写入控制 \overline{WE}) 和数据的输入输出(DQ)及字节控制 DQM 均与时钟同步,即根据命令控制的电平组合,相应的地址、命令和写入数据寄存器在时钟的上升沿接受输入信号;数据输出寄存器改变输出有效的输入命令(除无操作 \overline{NOP} 外的命令)只占用一个时钟周期。减少控制电路产生有效输入命令的周期数,可实现流水线操作,即输入操作命令与数据读出或写入可同时进行。由于同一行的存储单元在该行选中时,可同时读出到列读出放大器或由列写入数据寄存(驱动)器写入,所以可实现成组传送的读出或写入操作。即在输入第一个地址后,读出或写入数个行地址相同,但列地址连续或等距变化的存储单元的数据。后续地址不需由芯片外输入,而由列地址计数器在输入的的第一个列地址的基础上,按加电后设置的成组传送地址生成规则产生。后续地址存储单元的读出时间,只是输出寄存

器相对于时钟信号的延迟。数据的成组传送可提高系统的总体性能,以满足高速数字系统的要求。

工作过程和主要命令 SDRAM 芯片在加电后必须执行初始化流程。主要执行设置模式寄存器命令(MRS),设置成组操作长度及地址变化规则、列选等待时间(CL)等用于 SDRAM 芯片和存储器控制器芯片配合的关键参数。

数据读写过程如下。首先执行打开存储体和页面的命令(ACT),输入体地址和行地址(即页面地址)。体地址 BA 用于选存储体。对选中的体,行地址经行地址译码与驱动电路将它选中的一条行线驱动到高电平,使接在选中行线的 MOS 管全部导通。因选中行存储单元的电容与相应的列线相连,存储电容上的电荷与列线寄生电容的电荷重新分配,使列线电平变化,所存数据也受到破坏。由于存 0 与存 1 列线电平差别小,为了便于鉴别,读出放大电路采用差分放大。然后再鉴别所存储的数据是 1 或 0,并将其存入选中体各列的数据寄存器中。再按所存入的数据驱动列线,通过选中行的 MOS 管对电容充电(存入 1)或放电(存入 0)。这样将数据重新写入存储单元(恢复)。

ACT 命令后,延迟 t_{RCD} ,输入读出或写入(READ/WRITE)命令,同时输入列地址。时钟的上升沿将列地址打入列地址缓冲。经列地址译码,读出时经 CL-1 个时钟周期延迟,时钟上升沿将选中的列数据寄存器中的数据送到输出数据缓冲寄存器输出。成组传送时,连续输出由列地址计数器产生的后续列地址确定的列数据寄存器中的数据。直到输出的数据数为所设置的成组传送长度或下一操作命令中断了成组传送。写入时,与输入列地址同时,输入第一个写入数据及其控制 DQM。写入数据经输入数据缓冲电路送入由列地址选中的列线的数据寄存器。改变了由行选中读出送到该寄存器的数据内容。写入的数据与其他列未选中由存储单元读出的数据一起写入到行线选中的存储单元中。后续时钟上升沿接受的写入数据和 DQM,送到根据列地址计数器产生的后续列地址所选中的列数据寄存器中,从而改变在选中行上列地址也选中的存储单元的数据,直到接受的数据数为所设置的成组传送长度或下一操作命令中断了成组传送。

在执行新的选体和选行命令前,必须先执行预充电(PRE)命令,即关闭存储体和页面。使选中行变为未选状态,列线电平恢复到行线选中读出前的稳定状态,以保证新的选行命令读出数据的正确性。

可对 BA_0, BA_1 选择的体预充电,也可对所有体预充电。在输入列地址时, A_{10} 脚不作列地址用,而作自动预充电(AP)控制输入。AP 为高电平时,按内部时序要求,在执行读写命令后自动执行选中体的预充电操作。

刷新 刷新地址计数器可提供执行刷新操作时的行地址。非同步 DRAM 芯片有三种刷新方式:仅 RAS 刷新方式, $\overline{\text{CAS}}$ 在前的刷新方式(有控制内部刷新地址计数器的计数操作),隐含方式(此方式类似于 $\overline{\text{CAS}}$ 在前的刷新方式,但与读写操作合并进行,也可用内部刷新地址计数器确定刷新地址)。SDRAM 提供两种刷新方式:自动刷新和自刷新。自动刷新用于有时钟输入的场合,通过输入自动刷新命令(CBR),刷新内部刷新地址计数器确定的一行存储单元,并使计数器加 1。自刷新在无时钟输入时使用。先执行进入自刷新命令(REFS-EN)进入自刷新模式。开始内部刷新地址计数器计数和刷新操作。刷新控制电路按一定的时间间隔提供刷新控制信号,以保证各行相邻两次刷新的间隔时间在规定的范围内,以保证低功耗待机时,所存数据不丢失。系统工作时必须退出自刷新状态。最后执行退出自刷新命令(REFS-EX)。

刷新间隔时间通常以需刷新的行数(即刷新周期数)和每行连续两次刷新的最大间隔时间表示。例如 256Mb SDRAM 芯片可以集中刷新,每隔 64 ms,连续对 8 192 行刷新。也可分散刷新,即每隔 7.8 μs 刷新一行,每行连续两次的刷新间隔时间不超过 64 ms。

存取速度 这是表征 DRAM 芯片性能的一项指标。对非同步 DRAM 芯片,通常用行地址读出时间 t_{RAC} 表示,即由 RAS 下降边到读出数据有效的时间,一般为 60 ns, 50 ns, 45 ns。实际使用时,列地址读出时间 t_{CAC} 也很重要,它是由 $\overline{\text{CAS}}$ 下降边到读出数据有效的时间,与 t_{RAC} 对应的 t_{CAC} 为 15 ns, 13 ns, 12 ns。列选读出时间比行选读出时间小得多。对 SDRAM 芯片通常用时钟频率表示。SDRAM 芯片加快了连续访问相同行地址的后续列地址存储单元的访问时间,但并未加快选中行第一个列地址存储单元的访问时间。行选读出时间的改进主要靠 CMOS 工艺的改进。

DDR SDRAM 芯片在 SDRAM 基础上增加了数据的字节选择信号 DQS。数据输入输出以 DQS 的上升下降边为参考。写入时 DQS 为输入,读出时 DQS 为输出。其数据速率是 SDRAM 芯片的数据速

率的2倍。DDR2 SDRAM 芯片在 DDR SDRAM 芯片的基础上,将时钟频率增加一倍,数据速率是 SDRAM 芯片的数据速率的4倍。Rambus DRAM (RDRAM) 芯片是一种 SDRAM 芯片,它与普通 SDRAM 芯片的最主要差别是提高时钟频率,减少数据、地址和控制信号的线数,每种线都用4个时钟周期传输8组数据(数据包传输)构成完整的数据、地址和控制信号。由于时钟和信号传输频率的提高,对信号传输完整性的要求也大大提高。由此带来接口逻辑、匹配方式、走线要求及时钟线的变化。为满足特殊应用的要求,也有以 DRAM 芯片存储单元阵列列为内核,输入输出数据速率为 SDR/DDR 的专用 SDRAM 芯片,如图像存储芯片、网络存储器芯片和移动通信 SDRAM 芯片等。

为适应数字系统对存储容量和字长的需要,按字长以各种 DRAM 芯片为基础,设计了标准的单列存储器模组(SIMM)、双列存储器模组(DIMM)和 RDRAM 存储器模组(RIMM),以便使用者根据需要选择或扩展容量。

DRAM 芯片的工作电压向低电压发展,以降低功耗。高速和高密度(大容量)工艺本身的发展允许的最高工作电压也相应降低。非同步 DRAM 芯片工作电压为 5V,SDRAM 芯片工作电压 3.3 V,DDR SDRAM 芯片工作电压为 2.5 V,DDR2 SDRAM 芯片工作电压为 1.8 V。

参考文献

1. Rabaey J M. Digital Integrated Circuits: A Design Perspective. Prentice Hall, 1996. 北京:清华大学出版社,1998(影印版)

2. <http://www.samsung/Products/Semiconductor>
(孙祖希)

duanyu jieyou wenfa

短语结构文法 (phrase structure grammar)

形式语言理论中的一种重要文法。一个四元组 $G = (\Sigma, V, S, P)$, 其中 Σ 是终结符的有限字母表, V 是非终结符的有限字母表, $S (\in V)$ 是开始符号, P 是生成式的有限非空集, P 中的生成式都为 $\alpha \rightarrow \beta$ 的形式, 这里 $\alpha \in (\Sigma \cup V)^* V (\Sigma \cup V)^*$, $\beta \in (\Sigma \cup V)^*$ 。短语结构文法又称为 0 型文法。因对 α 和 β 不加任何限制, 故也称其为无限制文法。0 型文法生成的语言类与图灵机接受的语言类相同, 称为 0 型语言类(常用 \mathcal{L}_0 表示)或递归可枚举语言类(常用 \mathcal{L}_e 表示)。

例如: $G = (\{a\}, \{[,], A, D, S\}, S, P)$, 其中:

$P = \{S \rightarrow [A], [\rightarrow [D, D] \rightarrow], DA \rightarrow AAD, [\rightarrow \wedge,] \rightarrow \wedge, A \rightarrow a\}$, 显然, G 是短语结构文法, 它所生成的语言 $L(G) = \{a^{2^n} | n \geq 0\}$ 是 0 型语言。

0 型语言在一些代数运算下的封闭性如表 1 所示, 关于判定问题的一些结果如表 2 所示。表中 D 表示可判定, U 表示不可判定, G 表示文法, L 表示语言。

表 1 $\mathcal{L}_0, \mathcal{L}_1$ 在代数运算下的封闭性

语言类	\mathcal{L}_0	\mathcal{L}_1
代数运算		
求并	✓	✓
求连结	✓	✓
求闭包	✓	✓
求补	×	?
求交	✓	✓
与正则语言相交	✓	✓
反演	✓	✓
置换	✓	×

注: ✓ 表示封闭 × 表示不封闭 ? 尚未解决

表 2 与 $\mathcal{L}_0, \mathcal{L}_1$ 有关的判定问题

语言类	\mathcal{L}_0	\mathcal{L}_1
判定问题		
任意 $x \in L(G)$?	U	D
$L(G_1) \subset L(G_2)$?	U	U
$L(G_1) = L(G_2)$?	U	U
$L(G) = \emptyset$?	U	U
$L(G)$ = 无限集合?	U	U
$L(G) = \Sigma^*$?	U	U

短语结构文法的标准型为: $A \rightarrow \xi, A \rightarrow BC, A \rightarrow \wedge, AB \rightarrow CD$, 其中 $\xi \in (\Sigma \cup V)$, $A, B, C, D \in V$, \wedge 是空字。

参考文献

Hopcroft J E, Ullman J D. Introduction to Automata Theory, Languages and Computation. Addison-Wesley Publishing Company, 1979 (马世骅)

duanyu jieyou yufa

短语结构语法 (phrase structure grammar, PSG)

作为形式语言理论中的一种主要文法, 它是乔姆斯基层次中心的 0 型文法(参见短语结构文法); 作为现代语言学的一种语法理论, 它特指 N. 乔姆斯基 1957 年在论文《句法结构》中提出的一种生成语法模式, 是转换生成语法模型中句法部分的基本部件, 其规则形式属上下无关文法, 即 2 型文法。

比如,描写英语简单句的几条短语结构语法规则是:

S(句子) → NP(名词短语) VP(动词短语)

NP(名词短语) → T(限定词) N(名词)

VP(动词短语) → V(动词) NP(名词短语)

于是句子:

The cat chases the mouse.
T N V T N
猫 追逐 老鼠

经过上述规则的推导,可获得如图1所示的句法分析结果。

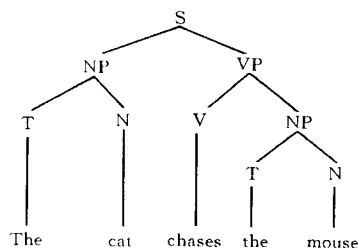


图1 句法树

参考文献

1. 石纯一,黄昌宁,王家庶. 人工智能原理. 北京:清华大学出版社,1993
2. Noam Chomsky N 著. 句法结构. 邢公畹等译. 北京:中国社会科学出版社,1979 (黄昌宁)

duixiang

对象(object) 面向对象系统中运行时刻的基本成分,它是数据和操作(或谓属性和行为)的封装通信单位。数据表示对象的属性状态;操作(或称方法)决定了对象的行为和与其他对象进行通信的接口。数据是对象私有的,只有该对象内的操作才能进行存取。

任何事物均有各自的属性与行为,当考察其某些属性与行为并进行研究时,它便成为有意义的对象。当用面向对象的方法进行计算机模拟时,应区别两种不同含义的对象:问题对象和计算机对象。前者是指现实世界中存在的实体在问题域中的抽象,后者是指问题对象在计算机系统上的表示。

面向对象语言中的对象指的是计算机对象。对象具有被动和主动两个方面。被动方面指的是其相对静态属性,借以认识对象,并对之归类。主动方面指的是其具有改变静态属性的动态行为。静态属性与动态行为相互影响,属性决定行为,行为改变属

性。此外,对象并非孤立存在,彼此之间通过传递消息进行交互。这样,对象可表示成三元组(接口、状态、操作)。动态地看,对象又是通信自动机。

对象内部有两类操作:一类是改变其内部属性的操作,另一类是生成输出的操作。

对面向对象语言中的对象,流行的观点有两种:一是以 **Smalltalk** 语言为代表的广义解释,将对象定义为计算机系统的所有成分,强调动态系统本身;另一是以 **Eiffel** 语言为代表的狭义解释,将对象定义为计算机系统中用以模拟问题对象的成分,着重考察系统和现实世界间的对应关系。

计算机对象具有5个基本特性:①自治性,指对象具有一定的独立计算能力;②封闭性,指对象具有信息隐蔽的能力;③通信性,指对象具有与其他对象通信的能力;④被动性,指对象的状态转换由外部刺激引发;⑤暂存性,指对象的动态创建与消亡。

参考文献

1. 徐家福等. 对象式程序设计语言. 南京:南京大学出版社,1992
2. Special Issue on Object-Oriented Design. CACM,1990,33(9) (张家重 王志坚)

duixiang-guanxi shujukuku

对象-关系数据库(object-relation database)

支持对象特征的关系数据库。对象-关系数据库基于对象-关系数据模型。该模型通过提供处理复杂对象的数据类型扩充关系模型。

20世纪80年代中期,为了满足新一代数据库应用的需求,面向对象数据库系统的研究与开发应运而生。在面向对象技术与数据库技术相结合的过程中,主要是沿着两条技术路线展开的。一条是建立纯粹的面向对象数据库管理系统(OODBMS),支持面向对象数据模型。另一条是以关系数据库和结构查询语言(SQL)为基础加以扩展,增加面向对象特性,建立对象-关系数据库管理系统(ORDBMS),简称对象-关系数据库系统。

面向对象数据库系统对一些特定的应用领域(例如计算机辅助设计)能较好地满足其需求。但是纯粹的面向对象数据库系统不支持结构查询语言(SQL),在通用性方面丢失了关系数据库的优势。1990年高级DBMS功能委员会发表了“第三代数据库系统宣言”,提出了面向对象数据库系统必须满足的两个条件:①支持核心的面向对象数据模型的

主要特征;②支持传统数据库系统所有的数据库特征。

也就是说,第三代数据库系统必须保持传统数据库系统的非过程化数据存取方式和数据独立性,应该继承数据库系统已有的技术,不仅能很好地支持对象管理和规则管理,而且能更好地支持原有的数据管理。对象-关系数据库系统就是按照这样的目标,将面向对象技术与关系数据库技术结合起来。

对象-关系数据库系统除了具有原来关系数据库的各种特点外,还应具有以下特点:

(1) 扩充数据类型 传统的关系数据库管理系统(RDBMS)只支持某一固定的类型集,不能依据某一应用所需的特定数据类型来扩展其类型集。对象-关系数据库系统允许用户在关系数据库系统中扩充数据类型,即允许用户根据应用需求自己定义数据类型、函数和操作符。

(2) 支持复杂对象 能在SQL中支持复杂对象。复杂对象是指由多种基本数据类型或用户自定义的数据类型构成的对象。

(3) 支持继承概念 能支持子类、超类概念,支持继承概念,包括属性数据的继承和函数及过程的继承;支持单继承与多重继承;支持函数的一名多用(重载)(操作的一名多用(重载))。

(4) 提供通用的规则系统 能提供强大而通用的规则系统。规则在数据库系统及其应用中十分重要,在传统的RDBMS中用触发器来保证数据库数据的完整性。触发器可以看成规则的一种形式。对象-关系数据库系统要支持的规则系统将更加通用,更加灵活,并且与其他的对象-关系能力集成为一体,例如规则中的事件和动作可以是任意的SQL语句,可以使用用户自定义的函数,规则能够被继承等。这就大大增强了对对象-关系数据库的功能,使之具有主动数据库和知识库的特性。

由于对象-关系数据库系统既能适应新应用领域的需求也能继续满足传统数据库应用深化发展的需要,目前几乎所有的关系数据库厂商,都在不同程度上扩展了关系模型,推出了对象-关系数据库管理系统产品。对象-关系数据库系统获得了快速发展。结构查询语言(SQL)国际标准SQL3中也增加了对对象-关系数据库管理系统(ORDBMS)的特征。因此,对象-关系数据库系统的应用日趋广泛。

参考文献

1. 萨师煊,王珊. 数据库系统概论(第三版). 北京:高等教育出版社,2000

2. Stonbraker M M, Moore D. Object-Relational DBMSs: The Next Great Ware. Morgan Kaufmann Publisher, Inc., 1996(本书已由杨冬青,唐世渭,裴芳等译. 对象-关系数据库管理系统——下一个浪潮. 北京:北京大学出版社,1997) (王珊)

duochuliji xitong zongxian

多处理机系统总线 (multiprocessing system bus)

使多个处理机模块、存储器模块、输入输出部件以及网络接口等部件(或多个由中央处理器、局部存储器、输入输出部件所组成的计算机模块)实现互连的公共通路。这是组成多处理机系统的一种最简单最直接的结构形式。在多处理机系统总线中,欲传送的信息以分时或多路转换方式在系统的主控部件和受控部件之间传送。主控部件主要是处理机或计算机模块,受控部件可以是存储器模块或是I/O部件,也可以是处理机或计算机模块。

多处理机系统总线通常由在各种功能板上的系统总线、存储器总线和局部总线所组成,在CPU、存储器、I/O或网络接口板上都有局部总线。在存储器板上的局部总线称为存储器总线。典型的I/O总线是SCSI(小型计算机系统接口)或I/O通道,用来连接本地的磁盘或其他外部设备。设计多处理机系统总线的关键技术包括:总线仲裁、中断处理、协议转换、快速同步、高速缓冲存储器一致性协议、分离式的事务处理、总线桥接以及层次扩展等。

多处理机系统总线一般限制在较小的机架内,因此能由系统总线连接的处理机数就十分有限,为此常用层次式总线结构来提高其可扩展性。

总线互连方式的优点是:简单、价廉、易实现和增减模块灵活方便。因此早期的多处理机系统几乎全都采用总线互连,至今仍有不少多处理机系统部分使用总线结构。

总线互连的主要缺点是:一是可靠性差,易成为系统的单故障点,一旦总线失效将导致整个系统失效;二是带宽较窄,易成为系统的性能瓶颈。因为系统总线是共享介质网络,其带宽将为连接到总线的部件、模块和处理机所共享,因此可连接到总线的部件、模块和处理机的数量有限,过多的总线负载将导致主控器争用总线的加剧和信息传送时间的加长,从而使整个系统性能低下。多处理机系统总线通常以事务方式工作。简单的事务工作方式 of 连接式的,即在单一的事务操作中只能实现一个主控部件请求和一个受控部件的响应。复杂的事务工作方式

为分离式的,它将一个总线事务操作分成请求阶段(包括仲裁、请求和错误检查)以及应答阶段(包括完成、响应和传送数据),它允许总线先执行一个事务的请求阶段,然后在执行该事务的应答阶段之前执行另一个事务的请求阶段,从而以流水方式使用总线。这种工作方式可明显提高总线的带宽,但同时也将使一个事务操作的时延有所增加。

解决上述问题,通常采用以下两种方法:一是改善总线传输特性,用优质电缆组成总线,以提高总线传输速率;二是改变总线结构,增加处理机间的可能通信路径,减少竞争。具体的实现方案有3种:①重复设置多套总线,增加处理机之间可能利用的通信路径,从而成倍地加宽其有效传输频带;②按性质分别设置处理器总线、存储器总线、I/O总线等,再利用总线耦合器把它们连在一起;③采用分级总线结构。尽管总线结构在扩大系统规模上似乎有较大的局限性,但在新近发展的不少多处理机系统中,它仍然是获得广泛应用的一种主要结构形式,其原因一个是因为总线结构自身的改进;另一个则是受微处理器分布处理系统的影响。

多处理机系统总线通常由计算机厂商自行设计,因而是专用的,以获取高的带宽性能,如Sun公司的Gigaplane多处理机总线、SGI公司的POWER-path-2多处理机总线等。多处理机系统总线的发展趋向是标准化(参见**总线标准**)。已有一些总线标准支持多处理机系统。常用的有VME总线、Multibus-II总线和Futurebus等。VME总线以异步方式传送信息,Multibus-II总线则以同步方式传送信息。用总线结构互连多处理机时,应尽量采用支持多处理机系统的总线标准,如因需要必须采用专用总线时,也应通过总线适配器将它与标准总线相连。唯此各种按总线标准设计的外围设备才可方便地与多处理机系统相连。

多处理机系统总线的工作步骤为:首先,由各个需要使用总线的处理机或计算机模块争用总线使用权;其次,被选中的主控器和受控器与总线相连;最后,信息按一定规约通过总线在—对主控器和受控器间进行传送,必要时也可由一个主控器以广播方式向多个受控器发送。为了解决多个潜在主控器同时争用总线的问题,需由系统总线仲裁器进行裁决,以确定哪个请求源可使用系统总线。

在多处理机系统中,当有1个以上的处理机或计算机模块要求使用系统总线而导致争用时,由**系统总线仲裁器**来裁决该由哪个处理机或哪个计算机

模块使用系统总线。系统总线仲裁器一般由仲裁算法和相应的硬件构成。仲裁算法性能的优劣对系统性能有较大影响。

常用的仲裁算法有:

(1) 静态优先级算法 它为每个连到总线上的处理机(或计算机模块)分配一个惟一的固定优先级。当多个处理机同时请求使用系统总线时,仲裁器使优先级最高的申请者使用总线。常用菊花链方式来确定优先级,越靠近仲裁器的处理机的优先级越高。这种算法的优点是简单、易实现,缺点是优先级低的处理机很少有机会使用总线。

(2) 均等算法 它通常以轮询方式将总线按固定长短的时间片依次供各处理机使用。常用于同步总线。优点是算法较简单且能保证各处理机有均等机会使用总线。缺点是平均等待时间较长。此外,若轮到的处理机不用总线时,将造成总线带宽的浪费。

(3) 动态优先算法 它根据总线使用情况和相应规则,动态地改变连接到总线上的处理机的优先级。如近期最少使用算法,将最高的优先级分给在最长时间内未使用总线的处理机。又如循环菊花链算法,根据离最后一次使用总线的处理机所处位置远近来分配优先级。距离越近的处理机,它的优先级越高。动态优先算法的优点是兼顾了前两种算法的优缺点既有较小的平均等待时间而又可使系统中各处理机有更均等的机会使用总线,缺点是控制逻辑较复杂。

(4) 先来先服务算法 它不按优先级选择申请者,因而有较好的均等性,但实现较困难。主要作为一种衡量其他算法优劣的标准。

上述各种仲裁算法,可用集中式或分布式结构实现。集中式结构由一个仲裁器统一实现仲裁算法,常用轮流查询或独立请求和准用等硬件机构实现。分布式结构则将仲裁硬件分布到各个处理机中,分配给每个处理机一个惟一的优先号,欲请求使用总线的处理机将自己的优先号由各自的分布仲裁器送到共享的请求有效线上进行逻辑“或”操作,形成一个合成优先号。然后再由分布仲裁器将各处理机优先号与此合成优先号相比较,优先号小于此合成优先号的处理机将自动撤销请求,获得总线使用权的将是具有最高优先号的处理机。分布式仲裁结构的主要优点是具有较高可靠性。系统总线仲裁器的工作过程如下:首先通过请求线接收各处理机发来的使用总线请求;然后由仲裁器按照

仲裁算法加以裁决并向选中的处理机在总线准用线上发出总线有效信号;最后由被选中的处理机通过总线忙控制线,向其他处理机表明总线已被占用。主控处理机使用总线传送信息后便撤销总线忙信号,从而使仲裁器可再去响应和选择其他处理机对总线的请求。

标准总线都具有仲裁结构,如 VME 总线仲裁器采用集中式结构,而 Multibus II 和 Futurebus 等总线仲裁器则采用分布式结构。这些总线一般都支持优先和均等混合仲裁算法,以适应多处理机系统的需要。对外围部件使用优先仲裁算法,而对其他处理机则使用均等仲裁算法,以使各处理机有较均等的机会使用系统总线。

参考文献

1. 黄铠,徐志伟著.可扩展并行计算:技术、结构与编程.陆鑫达等译.北京:机械工业出版社,2000

2. Tanenbaum A S. Structured Computer Organization. 3rd Edition. New Jersey: Prentice Hall, 1990

(陆鑫达)

duodao chengxu sheji

多道程序设计 (multiprogramming) 系统同时接纳多个程序进入内存让它们在操作系统控制下交迭或夹插执行的方式。

20 世纪 60 年代初计算机系统中的处理器比外部设备操作的速度要快得多,同时外部设备也能独立地和各种处理器并行地工作。如果仍和过去一样每次只接纳一道程序来执行,那么计算机系统的效率就很低。因为当一道程序启动外部设备后,在外部设备操作结束前程序往往不能继续执行下去而处于等待状态时,尽管处理器可以工作,但没有工作可做。如果系统让若干道程序同时进入内存,当一道程序因为启动外部设备或其他原因暂时不能继续执行而处于等待时,系统可让另一道等待执行的程序来运行。这样做显然可以提高系统的效率。此外,从分时系统的角度看,由于多个用户同时分享计算机系统,如果让用户程序一道地串行执行,那么用户在终端就可能等待很长的时间,也就不能实现分时的要求了。

实现多道程序设计必须妥善地解决以下三个问题:存储浮动与程序保护,处理器管理和调度以及系统资源的管理。

由于编制程序时无法知道该程序与其他哪些程

序同时进入内存,当然也不知道它存放在内存何处,因此,程序中的地址必须是相对的,它可存于内存任何一个区域。此外,由于若干道程序同时存于内存,系统应有相应的保护措施保证各道程序之间互不干扰,特别地,系统必须防止一道程序的出错而影响另一道程序的执行(参见**存储管理程序**)。

处理器调度是实现多道程序设计的一项关键技术。当运行着的一道程序由于启动外部设备或其他原因而暂时不能执行下去时,操作系统的处理器管理就从等待运行的程序中选择一道执行(参见**处理器管理程序**)。

在多道程序设计系统中,系统资源为若干道程序共享,因此操作系统必须对共享的系统资源进行管理和调度。操作系统这项功能的主要内容是:分配和去配系统资源,协助使用系统共享资源,例如控制打印机进行打印输出,打开一个文件供用户使用等等(参见**文件管理程序**和**输入输出管理程序**)。为了提高系统效率,防止死锁发生,对资源管理和调度要设计好调度算法。

在多道程序设计概念的基础上形成了进程的概念。由进程组成的并发程序可以包括多道程序的概念,多道程序概念就包含于多进程概念之中了。

(孙钟秀)

duo fashe jiegou

多发射结构 (multiple issue architecture)

在 1 个机器周期内可发射多条指令给执行部件以提高指令执行并行性的处理机体系结构。

在先进的处理机体系结构中,一般采用流水线结构(参见**计算机流水线**)以提高指令执行的并行性。自**精简指令集计算机 (RISC)**推广以后,各种处理机都力图在流水线的基础上把指令系统中的基本指令用 1 个机器周期执行完毕。这时,普遍用 CPI(执行每条指令的平均周期数)来衡量处理机的指令执行并行性。由于指令系统中除了基本指令以外,还有一些较复杂的指令,它们不可能在 1 个机器周期内执行完毕。因此,在每个机器周期内只发射 1 条指令的处理机体系结构中(这种结构称为**单发射结构**),CPI 的平均值只会大于 1。在设计优良的 RISC 单发射结构的处理机中,CPI 值可以达到 1.1~1.2。

为了进一步提高处理机指令执行的并行性,即在相同的时钟频率下进一步提高处理机的执行速度,必须减少 CPI。要使 CPI 值小于 1,则必须在流

流水线的基础上,在每个机器周期内发射多条指令给执行部件,这就是多发射体系结构。

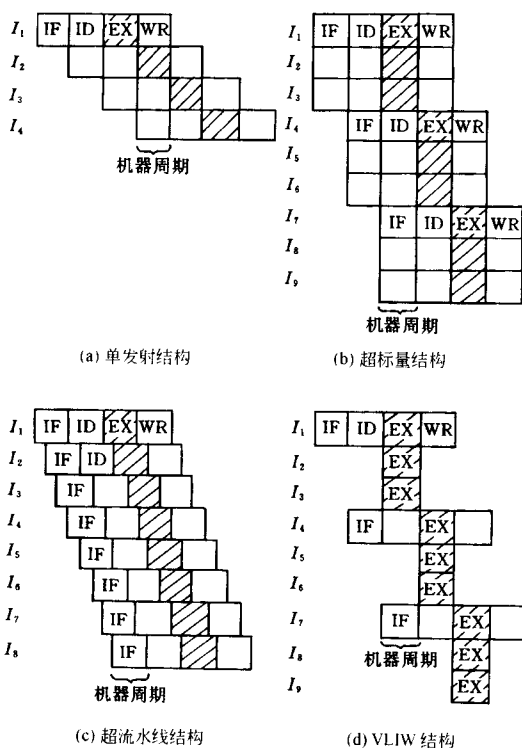
单发射和多发射结构处理机在流水线基础上的指令执行可以用图1表示。其中图1(a)是单发射结构处理机执行指令的流水线情况,流水线的4站:IF为取指令站,ID为指令译码站,EX为执行指令站,WR为写回结果站。图1(b),(c)和(d)分别是多发射结构的3种主要类型,即**超标量结构**、**超流水线结构**以及**超长指令字结构**。超标量结构是在每个机器周期内发射多条指令(图中为3条指令)。超流水线结构是把1个机器周期再细分为多个子周期,每个子周期发出1条指令,所以1个机器周期可发射多条指令。超长指令字结构是在取指令级IF内取出1条很长的指令字,这字中可以包含多条指令,如4条指令甚至几十条指令。这个长指令字在指令译码站ID一起译码,然后在执行站分别执行,图1(d)中是执行3条指令。多发射结构还有其他的类型,例如把超标量与超流水线结合起来,称为超

标量超流水线结构;又例如数据流计算机(参见**数据流计算机**),也是一种多发射结构。

超标量结构在发射出多条指令给执行部件以后,在执行站要同时执行这多条指令,所以要求处理机内具有多个执行部件,来完成不同类型的多条指令的同时执行。超流水线结构发出的多条指令,在执行部件中一条接一条地执行,但在每个子周期必须执行出结果,因此要求执行部件的工作周期较短。可以说,超标量结构是以空间换取时间,而超流水线是以时间换取空间。由于20世纪90年代初的半导体芯片上已可以集成几百万个晶体管,所以当时的大部分多发射结构都是超标量结构。后来VLSI电路的线宽已降到0.5 μm 以下,芯片的工作电压降为3.3 V,这些都有利于芯片工作频率的提高,因此90年代中期,多发射结构已采用超标量超流水线结构。超长指令字结构可开发更多的指令级并行性,但由于其指令格式较难与已有的计算机做到二进制兼容,另外,编译优化的难度也较大,所以超长指令字多发射结构未能被广泛采用。

多发射结构的目的是提高指令执行的并行度,即在维持与已有处理机二进制兼容的情况下使处理机的处理速度升级。多发射结构的出现,对于编译优化技术的要求也提高了,尤其是要解决指令级的转移相关性与数据相关性问题。这些相关性问题使指令执行并行性的提高受到约束。为了解决这一瓶颈,需要发展编译优化技术与指令级并行处理技术。

(李三立)



IF —— 取指令; ID —— 指令译码;
EX —— 执行指令; WR —— 写回结果

图1 单发射结构和多发射结构的指令执行流水线

duo fenbianlü zaoxing

多分辨率造型 (multi-resolution modeling)

在几何造型的基础上,对同一场景或场景中的物体生成不同细节表示的造型方法,常用于生成多分辨率模型。

多分辨率模型对同一个场景中的不同物体或同一物体的不同部分,采用多种不同细节的描述,它是控制场景复杂度、加速图形绘制、提高交互性的一个非常有效的方法。从多分辨率模型表示应能生成多个不同的视图,从而适应交互观察的需要。

目前提出的多分辨率模型的生成算法主要有两种:一种是基于小波的多分辨率分析,小波技术为多分辨率分析提供了一种简单、一致的方法来处理复杂的网格在存储、传输、绘制以及编辑处理中的问题;另一种方法是基于网格简化或者网格细分操作的多分辨率造型。基于网格简化或网格细分多分辨

模型一般采用层次或者序列结构组织网格简化或细分中生成的数据。以网格简化为例,层次结构将每次简化操作新生成的图元作为父结点,将被删除的图元作为子结点,从而将数据组织成一棵层状的树;序列结构将所有的数据组织成序列,排在后面的记录是从前面的记录中简化而来的。

递进网格(PM)是一种多分辨率模型表示,采用树结构记录新生成的结点与被删除的结点之间的父子关系,从而支持有选择的精化和简化操作。另外一种多分辨率表示方法是黏合多分辨率模型(GMM),GMM把在三维空间内有邻接关系、出现在层次树的不同层上的三角形编码在一起,以便快速地访问拓扑结构上相邻的三角形和在简化过程中相邻的三角形,为任意区域中网格的简化和精化提供了统一的操作模型。

在基于多分辨率模型的绘制算法中,如果一个物体离视点较远,或者这个物体较小,就可以用较低分辨率的模型进行绘制;反之,如果一个物体离视点较近,或者物体较大,就必须用较高分辨率的模型进行绘制。多分辨率模型可用于支持实时绘制、几何模型压缩和传输,在虚拟现实、交互式可视化、飞行仿真等系统中取得了成功的应用。

参考文献

石教英等. 虚拟现实基础及实用算法. 北京: 科学出版社, 2002 (潘志庚)

duolei daishu

多类代数 (many-sorted algebra) 在通常的代数中引进类型符号而形成的代数理论。最初由 Bénabou 于 1968 年以范畴论的形式提出。现在,多类代数已成为形式语义学的主要理论工具,它在软件规约、程序正确性证明等领域有重要应用。

若 S 为类型符号集合, Σ 为运算符集合, 并且 Σ 中每个运算符的类型形如 $s_1, \dots, s_n \rightarrow s_0$ ($n \geq 0$), $s_i \in S$, 则称 $\langle S, \Sigma \rangle$ 为基调。设 $A = \{A_s \mid s \in S\}$ 为族, F 为函数族, 如果对 Σ 中任意 $\sigma: s_1, \dots, s_n \rightarrow s$ ($n \geq 0$), 都有 F 中惟一的函数 $\sigma_A: A_{s_1} \times \dots \times A_{s_n} \rightarrow A_s$, 则称 $\langle A, F \rangle$ 为一个 Σ -代数。所有 Σ -代数的全体构成多类代数族。

给定基调 $\langle S, \Sigma \rangle$ 及 Σ -代数 $\langle A, F \rangle$ 和 $\langle A', f' \rangle$ 。从 $\langle A, F \rangle$ 到 $\langle A', f' \rangle$ 的 Σ -同态是满足以下条件的函数族 $\{h_s: A_s \rightarrow A'_s \mid s \in S\}$: 对 Σ 中任意运算符 $\sigma: s_1, \dots, s_n \rightarrow s$ ($n \geq 0$), 有 $h_s(\sigma_A(x_1, \dots, x_n)) =$

$\sigma_{A'}(h_{s_1}(x_1), \dots, h_{s_n}(x_n))$, $x_i \in A_{s_i}, \dots, x_n \in A_{s_n}$ 。由全体 Σ -代数和 Σ -同态构成的范畴的初始对象和终止对象分别称为初始代数和终止代数。

Σ -代数 $\langle A, F \rangle$ 上的同余关系 \equiv 为满足下述条件的关系族 $\{\equiv_s \mid s \in S\}$: ① 对任意 $s \in S$, \equiv_s 是 A_s 上的等价关系; ② 若 $a_i \equiv_{s_i} a'_i$, 则对 Σ 中任意 $\sigma: s_1, \dots, s_n \rightarrow s$ ($n \geq 0$), 皆有 $\sigma_A(a_1, \dots, a_n) \equiv_s \sigma_A(a'_1, \dots, a'_n)$ 。基于同余关系 \equiv 可得到 $\langle A, F \rangle$ 的商代数 $A/\equiv = \langle \{A_s/\equiv_s \mid s \in S\}, F/\equiv \rangle$, F/\equiv 中的函数定义如下: 对 Σ 中任意 $\sigma: s_1, \dots, s_n \rightarrow s$ ($n \geq 0$) 及 $a_i \in A_{s_i}$, 有 $\sigma_{A/\equiv}([a_1]_{\equiv_{s_1}}, \dots, [a_n]_{\equiv_{s_n}}) = [\sigma_A(a_1, \dots, a_n)]_{\equiv_s}$ 。

给定基调 $\langle S, \Sigma \rangle$ 及变元集族 $X = \{X_s \mid s \in S\}$ 。基于 X 的自由生成 Σ -代数 $T_\Sigma(X) = \{T_{\Sigma,s}(X) \mid s \in S\}$ 构造如下: ① $X_s \subseteq T_{\Sigma,s}(X)$ 且对 Σ 中任意 $\sigma: s_1, \dots, s_n \rightarrow s$ ($n \geq 0$), 若 $t_i \in T_{\Sigma,s_i}(X)$, 则 $\sigma(t_1, \dots, t_n) \in T_{\Sigma,s}(X)$; ② 对 Σ 中任意 $\sigma: s_1, \dots, s_n \rightarrow s$ ($n \geq 0$) 及 $t_i \in T_{\Sigma,s_i}(X)$, $\sigma_{T_\Sigma(X)}(t_1, \dots, t_n) = \sigma(t_1, \dots, t_n)$ 。特别地, 若 $X = \emptyset$, 则记 $T_\Sigma(X)$ 为 T_Σ , 称为 Σ -项代数或自由字代数。对任意函数 $h: X \rightarrow A$, 存在惟一的 Σ -同态 $\bar{h}: T_\Sigma(X) \rightarrow A$ 扩充 h , 也即 $\bar{h}(x) = h(x)$ ($x \in X$)。

抽象数据类型的语法形式为 $\langle S, \Sigma, E \rangle$, 其中 E 为 $T_\Sigma(X)$ 中的项所形成的等式或条件等式公理。满足 E 的 Σ -代数称为 $\langle S, \Sigma, E \rangle$ 的语义模型。所有模型所构成的范畴的初始对象和终止对象分别称为 $\langle S, \Sigma, E \rangle$ 的初始语义和终止语义。前述 T_Σ 的重要意义在于: 经由 E 导出的某些同余关系的划分得到的 T_Σ 的商代数可以构成 $\langle S, \Sigma, E \rangle$ 的初始语义和终止语义。在初始语义中成立的许多性质可推广至抽象数据类型的其他语义模型。

为了以结构化方式描述抽象数据类型, 需要引进层次协调性和充分完全性。对于抽象数据类型 $ADT_i = \langle S_i, \Sigma_i, E_i \rangle$ ($i = 1, 2$), 如果 $S_1 \subseteq S_2$, $\Sigma_1 \subseteq \Sigma_2$, $E_1 \subseteq E_2$, 则称 ADT_2 是 ADT_1 的扩充。 ADT_2 相对于 ADT_1 的层次协调性是指: 对 ADT_1 的任意项 $t, t', t = t'$ 在 ADT_1 中可证当且仅当 $t = t'$ 在 ADT_2 中可证。充分完全性是指: 对 ADT_2 中任意项 t_2 , 如果 t_2 的类型在 S_1 中, 那么必存在 ADT_1 中的项 t_1 , 使得 $t_1 = t_2$ 在 ADT_2 中可证。为了保证语义上的合理性, 在对抽象数据类型进行扩充或组合时, 必须考虑层次协调性和充分完全性。

为了处理计算机科学和软件工程中的复杂问题, 人们对多类代数进行了各种扩充, 以便处理部分

函数、高阶函数、异常机制、子类型机制、非确定性和并发性。例如,序类代数就是在多类代数的类型符号集合上引进偏序关系得到的代数理论。

参考文献

陆汝钊. 计算机语言的形式语义. 北京: 科学出版社, 1992 (谭庆平)

duolu fuyong

多路复用 (multiplexing) 通过同时携带多个传输信号来高效率地使用传输介质的方式。多路复用(multiplex)这一术语源于拉丁词 multi 和 plex 的组合。多路复用使用多路复用器连接许多低速线路,并将其各自所需的传输容量组合在一起后,在一条速度较高的线路上传输。即仅由一条较高速度的线路传输所有的信息,而不是在每一个发送端和每一个接收端之间连接着许多低速线路。

图1说明了多路复用器的工作原理。多路复用器有 n 个输入端(n 为任意正整数,大小取决于所用传输介质的限制)。多路复用器直接与传输介质连接,而数据通过传输介质传送出去。传输介质和另一端的多路复接器相连。这种传输介质是一条传送 n 个独立子信道的电路。多路复接器接收多路复用信号,并用子信道将其分开,然后把它们送到适当的输出端。不论一个多路复用系统中输入或输出端的数量多少,都只需要一条传输线路。

在一条传输线路上多路传输许多信号有下列优点:①仅需一条传输线路 所需的传输介质较少,所用的传输介质的容量可以得到充分利用。②降低

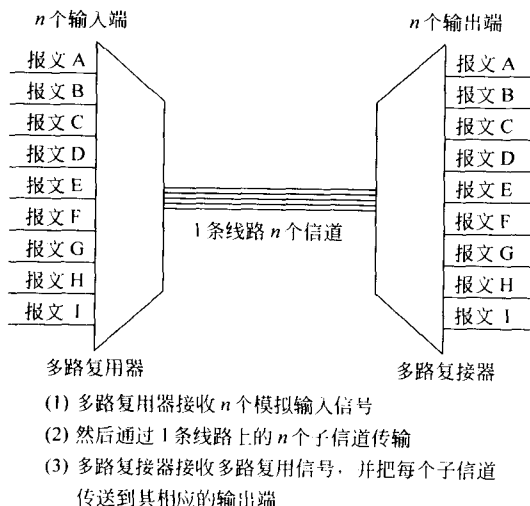


图1 多路复用

了设备费用 通信系统的费用因省掉不必要的传输线路而减少。③多路复用系统对用户是透明的,工作效率高。

配置多路复用线路有许多种不同方法,多路复用器的类型也各异,包括频分多路复用、时分多路复用、波分多路复用以及统计时分复用等。

参考文献

胡道元. 计算机网络(高级). 北京: 清华大学出版社, 1999 (胡道元)

duomeiti jishu

多媒体技术 (multimedia technology) 为了把文本、图形、图像、声音、动画、视频等多种形式不同而逻辑上相关的内容向用户进行展现而对它们进行综合处理的一种技术。“多媒体”是形容词,一般应与名词联用。例如,具有上述特性的计算机就是多媒体计算机,具有上述功能的通信系统就是多媒体通信系统,能够有效地储存、管理、检索多媒体信息的数据库系统就是多媒体数据库系统。

多媒体技术强调的是交互式综合处理多种信息媒体的技术。从本质上来说,它具有3种重要的特性:

第一是信息媒体的多样性,它不仅能处理传统的数值、文字、静止图像(静态媒体),更重要的是能有效地处理言语、音乐、动画和视频等随时间变化其内容的动态媒体;

第二是多种媒体的集成性(即综合性),它使用多种不同的信息媒体(特别是它能把静态媒体和动态媒体进行组合)综合地表现某个内容,取得更好的表现效果;

第三是内容展现过程中的交互性,在使用多媒体信息展现内容的过程中,用户可以对系统进行有效的、细粒度的控制,使人们获取和使用信息变被动为主动。

由于多媒体信息的数据量极大,特别是视频信息和音频信息,数据类型繁多,处理复杂,且往往有实时或限时处理的要求,不同的媒体相互间还必须提供同步机制等,因此多媒体技术比常规的信息处理技术要复杂得多。

多媒体技术是围绕着多媒体作品(也称为“多媒体文档”)展开的。一部数字影片、一段动画、一个计算机辅助教学(CAI)课件、一段远程教学的直播场景等都是多媒体作品。从用户角度看,多媒体信息处理的过程可分成图1所示的若干不同阶段,整个过程称为“媒体食物链”。

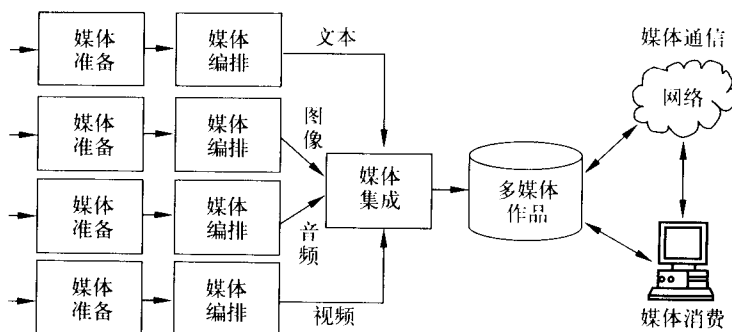


图1 多媒体信息处理过程

(1) 媒体准备 媒体准备由多媒体获取设备及相关软件完成,也可以人工创建。例如文字输入、图形绘制、**图像获取**、声音录制、视频拍摄等,这是把媒体引入计算机数字世界的关键。不同类型的媒体具有不同的性质,所使用的硬件设备及其处理软件各不相同,但目标相同:将媒体数字化或是人工创建数字媒体。

(2) 媒体编排 对各个媒体进行编辑和排版处理,如将不同的组成对象合成一体,修改其组成对象的内容(如字符、语句、配音、视频画面)及属性(如文字的字体、语句的速度和图像的颜色等)。这个阶段需要使用各种数字媒体所特有的编辑软件来完成,如文字处理软件、绘图软件、图像处理软件、声音编辑器、音序器软件、非线性视频编辑器等。

在媒体获取与编排的过程中,一个关键的问题是各种数字媒体如何表示(即媒体的编码)。数字媒体的编码的一个目的是为了压缩数据、提高编码效率,以减轻传输和储存的压力;另一个目的是为了合理地组织数据,以满足各种应用的要求,包括在各种不同系统中的交换和互操作。为此,国际标准化组织(ISO)、国际电工委员会(IEC)和国际电信联盟(ITU)等制定了一系列技术标准,如静止图像编码标准 JPEG, JPEG-2000;运动图像及其伴音的编码标准 MPEG-1, MPEG-2 和 MPEG-4 等。

(3) 媒体集成 为了把数字媒体对象相互关联构成一部多媒体作品,必须按作品脚本的规定正确处理各媒体对象相互间的时空关系。媒体集成涉及的技术问题很多,例如多媒体作品的数据库模型、规范与标准,媒体的空间布局与时间同步,媒体的说明、标识与描述,媒体的存储组织与管理,目录与索引,以及多媒体作品知识产权的管理与保护等。用于媒体集成的软件是多媒体写作工具软件。

(4) 媒体通信 媒体通信指在网络上传递、分发和交换多媒体作品,为最终用户提供远程的多媒体信息服务(例如视频会议、协同工作、电子邮件、信息检索等)。它既是技术手段,也是应用。为了进行多媒体数据通信,网络必须提供足够的带宽,满足规定的传输截止时间和端-端的抖动幅度等要求,提供不同的媒体数据流之间的同步机制,并能支持多播和广播方式的网络通信等,这就给通信技术和计算机网络带来了严峻的挑战。

(5) 媒体消费 媒体消费即多媒体作品的应用,通常有两种方式:单机应用(本地应用)方式和网络应用(分布式应用)方式。单机应用方式已经很普遍了,例如 CAI 课件、电子游戏、视频光盘(VCD)、各种光盘电子出版物等。网络多媒体应用是多媒体应用的主流,是发展方向,它技术难度大,但也是影响最大、最有效益的应用方式。网络多媒体应用又可以分成两大类型:①面向人与人通信的网络多媒体应用(p-to-p 方式) 此类应用过程至少有 2 个人介入,其目标是改善人们远程通信的效果。例如 IP 电话、实时远程教学、视频会议、网上电子游戏、多媒体电子邮件、多媒体文档交换、计算机支持的协同工作(CSCW)等。②面向人与系统通信的网络多媒体应用(p-to-s 方式) 此类应用的目标是改善或者提供创新的用户与信息源之间的通信方式,例如万维网(WWW)信息检索、网上购物、视频点播、网上电视(Web-TV)、网上广播(Web-radio)等。

多媒体技术是信息处理技术发展的一种必然,它的发展推动了许多相关产业的改造和调整,使计算机、通信、广播、出版、消费类电子产品等在技术上逐步走向融合。随着网络多媒体应用的飞速发展,人们的工作和生活方式将发生显著改变,多媒体技术也会得到进一步的发展。

参考文献

1. Ralf Steinmetz & Klara Nahrstedt. Multimedia Computing, Communications & Applications. 潘志庚等译. 北京: 清华大学出版社, 2000

2. 胡晓峰等. 多媒体技术教程. 北京: 人民邮电出版社, 2002 (张福炎)

duomeiti shujuku

多媒体数据库 (multimedia database) 以图形、图像、声音、视频、文本等多种信息载体的复合体为存储目标的数据库系统。它提供了对各种媒体数据的综合存储和查询能力。

同常规数据相比, 多媒体数据具有数据量大、类型多样, 具有实时性, 原始信息难以处理以及表现具有时、空特性等特点。

多媒体数据库具有广阔的应用前景, 目前已用于视频点播服务器、多媒体文本管理系统等领域。虽然多媒体数据类型多种多样, 应用各有特点, 但总体讲, 多媒体数据库系统由物理存储层、概念数据层、通信层、过滤层和用户界面层五部分构成, 如图 1 所示。

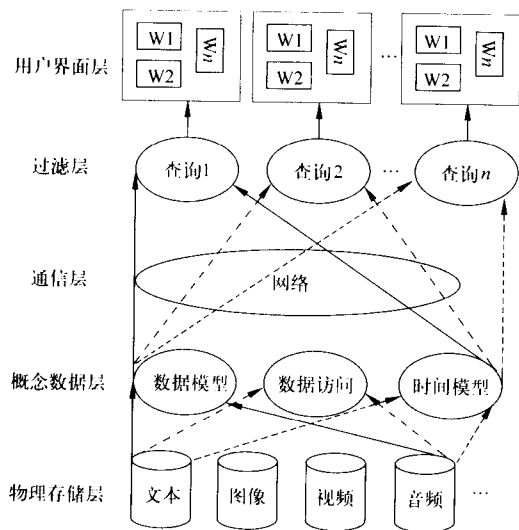


图 1 多媒体数据库系统结构

物理存储层主要解决多媒体数据的存储量和查询的带宽等问题, 多媒体数据的存储方法主要采用单硬盘存储、多硬盘存储、条带方式的多硬盘存储等方法。

概念数据层负责对多媒体数据的描述和查询。以常规数据类型构成的元数据在二进制多媒体数据的有效查询方面起到重要作用。元数据是描述数据及其环境的数据。元数据主要有两种, 一种是描述单一媒体的元数据, 包括对数据的特性、内容、来源、作者等信息的描述; 另一种是描述媒体间相互关系的元数据。目前多媒体数据特别是元数据的建模大多是采用面向对象模型。

对多媒体数据的查询包括除了针对元数据的查询以外还有针对多媒体数据本身的查询。对不同的媒体通常采用不同的访问技术。例如对文本数据采用: 全文扫描、反向文件、文本聚类等方法。对图像数据则包括: 对图像包含的对象和物理关系的查询以及对图像的某些物理特性的查询, 如颜色和纹理等。对视频数据包括: 对视频数据中对象和事件的查询、对摄像机的移动和对象的移动的查询等。

多媒体数据描述中的一个重要部分就是时间特性, 它将影响网络层的调度和用户界面的展示, 包括展示的起始时间、长度和同步关系。

多媒体对象多具有分布存储和数据量大的特点, 通信层提供了分布式计算环境下客户-服务器间的数据传输机制和服务质量保障机制。系统以用户的展示和查询调度为基础, 生成查询请求响应的服务质量参数并提供给网络服务模块, 用于对网络资源进行调度, 尽可能向用户提供最佳的服务质量。

在过滤层中, 用户可以定义对多媒体数据的查询。对多媒体数据的查询类型有多种, 包括: 对媒体信息内容的查询、基于实例的查询、时间索引查询、空间信息查询和应用特定的查询等。

在用户界面层中, 用户需要定义各种媒体在用户界面上的展示位置和方式, 以及各种媒体在展示时的同步关系, 即展示调度, 并将同步要求传送给通信层。

参考文献

1. Prabhakaran B. Multimedia Database management System. Kluwer Academic Publishers, 1997

2. Khoshafians, Balur A Brad. Multimedia and Imaging Databases. Morgau Kaufmann Publishers Inc., 1996

(汪卫 施伯乐)

duomeiti wendang

多媒体文档 (multimedia document) 包含多媒体数据的文档。它特指在媒体数量、媒体类型和媒体集成度 3 个测度上都有充分体现的文档。多

媒体文档要能够表示不同媒体数据的构造及其属性特征,能够指出不同媒体数据之间的相互关系,特别是反映高层应用语义的时空特性关系。

多媒体文档要体现:

(1) 数据多样性 多媒体文档必须支持多种数据类型,包括对字符文本、向量图形、位图图像、数字音频、动画、数字视频等与应用无关的通用数据类型的准确定义,和对与应用有关的数据类型提供建模工具及在演示时将它们映射到基本数据类型的说明。

(2) 同步特性 多媒体系统中存在着一个重要的问题是表现,这是多媒体系统所特有的,它不同于一般系统的显示,而是多种媒体数据的合成再现,加工再现,有交互性参与的创作再现。这其中同步问题是多媒体系统的一个重要特征,分布式多媒体系统中由于数据分布、网络资源有限造成的不确定性使同步问题更为突出。

(3) 交互特性 多媒体的一个关键特性是交互性,它将向用户提供更加有效的控制和使用信息的手段和方法,同时也为应用开辟了更加广阔的领域。交互可以做到自由地控制和干预信息的处理,增加对信息的注意力和理解。当引入交互后,活动本身作为一种媒体便介入了信息转变为知识的过程。媒体数据的简单检索与显示是多媒体的初级交互应用;通过交互特性使用户介入到信息的活动过程中,才达到交互应用的中级水平;当用户完全进入到一个与信息环境一体化的虚拟信息空间自由遨游时,才进入交互应用的高级阶段。

(4) 组织结构 文档包含文档结构和文档内容两方面的含义。文档结构描述文档中单个元素的联结关系或文档内容的组织方式。多媒体文档的组织结构要能反映其内容。超文本在高层上对结点和链的区分很适合表现文档结构的高层语义,直接反映了人类联想式思维的模式,是多媒体文档最为有效的组织形式。

(徐光佑 潘志庚 史元春)

duomeiti wendang guifan

多媒体文档规范 (specifications of multimedia document) 一些国际组织和企业联盟为便于多媒体文档的数据交换而制定的开放性的文档规范。常见的有 HTML, MHEG, VRML, SMIL 及 Hy-Time 等。

HTML (超文本置标语言) 标准通用置标语言 (SGML) 的一种独立于平台的格式定义,以置标 (说

明)来建立超文本、超媒体文档。HTML 于 20 世纪 80 年代末、90 年代初研制成功,目前是描述万维网 Web 页面的置标语言。HTML 文档是一种纯文本文件,可以用任何文本编辑器阅读和编辑。Web 页面中包含一系列 HTML 标记,以指示浏览器如何显示 Web 页面。当用户用浏览器与某个 Web 服务器建立连接时,Web 服务器通过网络把相应的 HTML 文件发送到用户浏览器。超文本置标语言 HTML 借助统一资源定位地址 (URL) 可以描述 Internet 中跨越结点的超链接,简单而实用地实现了以整个 Internet 空间为操作背景的超文本、超媒体的数据存取,且具有易于在不同表现系统上移植而保持文献的逻辑完整性的特点。

由于 HTML 主要描述的是页面的布局和外观,缺乏对内容语义的描述,另外,它的置标的集合是固定的,用户不能增加自己定义的置标。因此万维网联盟 (W3C) 于 1998 年初推出了被称为“第二代 Web 语言”的可扩展置标语言 (XML) 以解决 HTML 的固有问题,已逐渐被广泛接受。

MHEG ISO 多媒体与超媒体信息编码专家组的简称,也用以代表它制定的标准。该标准定义了结构化信息独立于系统的编码,用于存储、交换和执行多媒体演示。MHEG 遵从 OSI 的思想,基于 OSI 表示层的体系结构,使用 ASN.1 描述数据元素和数据结构,支持多媒体同步和用户交互,希望以此作为超媒体文档在不同系统间进行实时交换的工业标准,因而特别注重交互性和多媒体同步、实时表现、实时交换、最终形式表现等几方面。

MHEG 的基础就是被称为消息处理 (MH) 对象的信息的独立编码表示和信息的基本单元,以便在不同应用中处理和交换所用的这些对象。这些对象可以是简单的媒体对象,或者是带有内同步和链接的不同媒体的不同分量的复合对象。MHEG 在标准的设计中采用了面向对象的方法,但它对于标准的实施却并非必需。

MHEG 中提供了一种虚坐标系统,包括一个无限长的时间轴和长度有限的 X, Y, Z 3 个空间轴,用于规定内容对象在空间和时间上的定位和相互关系。在 MHEG 中,链接表示对象之间的关系,或者说是一个条件,当源对象的状态满足链接中定义的条件时,就让目的对象执行动作。所有的对象都靠这种方式联系起来,构成网状结构。

VRML (虚拟现实建模语言) 一种用于在 Web 上构作三维多媒体和共享虚拟世界的开放式语言。

VRML 的基本原理与 HTML 的基本原理一样简单,都是用一系列标记告诉浏览器如何显示一个文档,它们都是描述 Web 页面的描述语言。它与 HTML 不同的是,以 HTML 为核心的 Web 浏览器浏览的是二维世界,而以 VRML 为核心的 Web 浏览器浏览的是三维世界。在 VRML 中,以结点作为基本单位,将不同的结点以层次关系组织在一起,构成 VRML 中的场景图,使 Internet 用户犹如身处真实世界,在三维环境中随意探询 Internet 上丰富的信息资源。

SMIL(同步多媒体集成语言) 由万维网联盟(W3C)提出的一种语言规范,提供了一种统一的语言格式用以描述多媒体数据的集成和交互。SMIL 也是一种置标语言,它定义了一系列遵循可扩展置标语言(XML)规范的元素和属性,以生成包含多种媒体对象的复合多媒体文档,规定它们在时间和空间上的布局关系及运行环境等,SMIL 还提供了简单的基于事件的激活方式,以实现媒体对象的交互操作。

HyTime(基于时间的超媒体结构化语言) 它是 ISO 为指定超媒体和多媒体文件的逻辑结构而定义的标准。HyTime 主要研究多媒体同步的表示,超媒体在文档内或文档间的链接。HyTime 适用于综合的开放型多媒体和超媒体信息系统,以及在开放环境下的文档交换和操作管理。HyTime 并不规定信息内容的格式、编码或文件类型,而是提供一种框架结构,用置标语言来描述多媒体文档的超级链接方式,以明确多媒体信息内容在时间上和空间上的相互关系,从而实现同步。HyTime 与 MHEG 在很多方面是一致的,但它们的使用方法和应用环境却不同。涉及到文档的处理和交换可用 HyTime,涉及到对象的处理与交换则用 MHEG。同 MHEG 一样,HyTime 的文档实现也很复杂。

(徐光祜)

duomeiti zhuzuo gongju

多媒体著作工具 (multimedia authoring tool) 多媒体文档和多媒体应用程序的一种开发平台。它能够统一地编辑、管理多媒体数据,一般不需要高级语言编程就可以把这些数据连接成完整的多媒体文档或应用程序。

著作工具试图将多媒体应用开发过程中的编程工作简化,使得没有经验的用户也可以利用著作工具制作能展现图像、文本、动画、音频、视频等多媒体数据,并具有一定的交互能力的多媒体应用程序。著作工具的主要特点首先在于它简洁易用,通常在

几天,甚至几个小时之内,用户就可以开始进行创作。多数著作工具都提供图形界面,这些图形界面为用户设想出程序的隐含流向,并自动处理许多具体的编程指令。此外,著作工具通常都提供样板应用程序,这些样板只进行适当改动即可快速拼装一个应用程序。

多媒体著作工具的不足之处一般有两个:一是所开发的多媒体应用效率较低,通常要比用程序设计语言开发的多媒体应用运行速度慢;二是开发的多媒体应用的灵活性受到限制,有些特殊的或者复杂的功能著作工具并不提供。为了克服这些缺点,大多数多媒体著作工具都能支持一种或几种脚本语言,例如 VBScript, JavaScript 等,它们简单易懂,容易掌握。使用脚本语言编写的程序可以嵌在多媒体文本或应用中,从而方便地实现用户定制的功能。

多媒体著作工具作为一类特定用途的程序,并没有具体的设计标准,因此,目前可见到的千百种著作工具的著作方式多种多样,但归纳起来可分为下列 3 类:基于流程图的、基于卡片的、基于语言的。

①基于流程图的工具功能强大,但要求用户有相当的程序设计经验。②基于卡片的著作工具是按照超链接的结构设计的。超链接的结点由具有一定时空关系的多媒体数据构成,通常被看作卡片、页或场景,用户可直观地编辑卡片内的多媒体内容,操作直观而简便。③基于语言的著作工具为多媒体对象的操作设计了面向对象的操作语言,其语法容易理解,用户不用操心程序的细节,但要掌握这类语言,也需要较长时间的学习和培训。

多媒体著作工具生成的多媒体文档可以是私有的(自定义的文档格式),也可以是开放的,即遵循已有的公开的多媒体文档规范。

参考文献

1. Badgett T, Sandler C. Creating Multimedia on Your PC. John Wiley & Sons, 1994
2. <http://webopedia.internet.com/>

(徐光祜 张福炎)

duomotai renji jiaohu

多模态人机交互 (multimodal human-computer interaction) 通过对视频、音频、图像、手势等多种模态信息的综合处理,为用户提供自然、直观的人机交互方式的技术。多模态可以理解为多通道或多方式,是相对于键盘、鼠标、显示器等传统的单一的交互方式而言。

人与人之间自然的交互通过视觉、听觉、触觉、嗅觉甚至味觉等多种模态信息来完成。与此相对,目前的计算机用户主要使用键盘和鼠标等人工设备进行人与计算机的交互,这不利于使更多的非计算机专业人员使用计算机。多模态人机交互研究的目的是使计算机具有检测、感知和理解多种信息的能力,从而使用户能像人与人之间那样与计算机交互。

为使计算机具有感知能力,计算机或计算设备需要装备各种传感器和相应的识别、理解技术作为其输入系统,同时还需要装备相应的多媒体输出系统。对于视觉功能,需要装备摄像机,研究计算机视觉技术、人的情感理解、人的运动跟踪等。对于听觉功能,计算机需要装备拾音器,研究言语识别、言语合成、文字到言语转换的**文语转换**技术等。对于触觉需要有数据手套、操纵杆等触觉传感器和相应的信号处理和识别技术。

此外,多模态信息的融合也是重要的研究课题。在目前的系统中主要是二种模态信息的融合,如语音识别与笔输入、语音识别与唇读结合等。对人体的其他运动的识别也已引起了高度的重视,其中包括:跟踪和理解用户的注视方向和位置,头和身体的位置和姿态,脸部的表情,手势等。在基于生物特征的身份识别中已出现把多种行为(言语识别,手写体识别,人体运动识别)与生理特征(指纹,视网膜识别)利用传感器融合技术组合起来的趋势。

在人机接口中采用多模态人机交互技术还具有以下优点:①使不同模态的信息能相互补充和支持,从而使系统的性能比用单模态信息更为可靠。例如,把基于视觉信息的唇读识别与言语识别相结合,在高噪声环境下会显著提高单纯利用言语识别系统的性能。②使用户能根据变化的工作环境改变或切换不同的模式。

由于未来用户的终端设备已不只是桌面计算机,而将越来越多地使用笔记本电脑、PDA、移动电话等设备。其中多数设备没有键盘和鼠标,这使得多模态交互技术如笔输入、言语识别或其他的感知输入就成为必要。

近年来人-计算机交互(HCI)已扩展到人-机器交互和通信(HMC)。HMC研究的最后目标是使人与机器之间的通信就像是人与人之间的通信。此外,还要使机器能支持人与人之间的通信(例如,用于残疾人的接口)。这样,多模态人机交互技术就有了更为广泛的应用领域。

参考文献

1. 董士海,王坚,戴国忠等著. 人机交互和多通道用户界面. 北京:科学出版社,1999
2. Oviatt S. Multimodal Interface. In: Handbook of Human-Computer Interaction, (ed. By Jacko J & Sears A), Lawrence Erlbaum; New Jersey, 2002

(陶霖密 陈思义)

duomotai shengwu tezheng ronghe

多模态生物特征融合 (multi-modal fusion in biometrics) 综合考虑多种生物特征以提高识别性能的技术。

单个生物特征识别技术在鲁棒性、稳定性等方面还存在一定的问题;比如在噪声环境中利用语音的说话人识别系统将不能很好地运行(参见**说话人识别**),对于双胞胎,仅仅使用脸像特征也不能很好地区分(参见**人脸识别**),而对于**指纹识别**技术,有5%左右的人不能得到很好的指纹特征等。多模态生物特征融合技术,将多种生物特征联合以试图解决上述问题。比如考虑脸像、语音、指纹等多种生物特征的融合,当在噪声环境中语音特征不能很好发挥作用时,其他两个生物特征仍然能够得到很好的辨识效果。

多模态生物特征融合的研究内容主要包括:

(1) 融合的模态问题 即使使用哪些特征加以融合。事实上,几乎所有的生物特征都可以用来加以融合,不同的生物特征从不同的侧面反映了人的特性,因此将其通过一定的策略融合后均可以得到比单个特征更好的性能。语音和视觉特征由于说话时的必然相关性因而成为融合的首选。也有研究把指纹、脸像、语音等加以融合。图1给出了基于语音视觉特征的融合示意图,其中视觉特征又包括脸像和唇动两个方面。

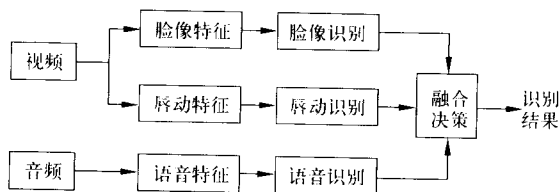


图1 基于脸像、唇动、语音的决策融合示意图

(2) 融合的层级问题 不同的生物特征可以在不同的级别上融合,例如:①**数据级融合** 对最底层的原始数据直接进行融合,因而包含了对原始特

征最充分、最有效的描述。然而由于数据间的强相关性、特征的复杂性等,使得该层级只能是理论上的存在,离实际应用还有较大的距离;②参数级融合

具有关联关系的不同的特征在参数提取及计算时往往具有相关性,通过参数级的融合,可以在计算某一特征参数的同时利用另一特征提供的信息;③特征级融合 输入数据经过前端处理后,分别得到每个生物特征的特征描述向量,在特征级融合阶段,将多个低维特征向量融合(合并)成高维的联合特征向量参数,并针对该高维特征向量建立模型,进行识别和决策;④模型级融合 考虑多个生物特征的关系,在此基础上建立联合模型,并基于此模型进行分类决策,在建立联合模型时,既可以考虑特征的关联,也可以考虑其区别特性;⑤决策级融合 具有不同模态的特征分别进行单模态的建模与识别,并将各自识别的中间结果参数,通过决策融合模块进行融合,最后通过多模态决策算法得到最终的鉴别结果。

上述各个层级融合的关系示意图2所示。

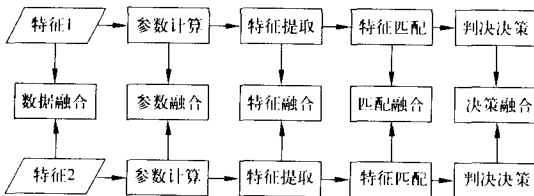


图2 多层次融合框架结构示意图

目前数据融合的研究主要集中在决策融合阶段,其优点是简单可行,不同模态的单个特征可以分别独立处理,得到单特征匹配结果,然后通过决策融合算法加以综合得到最终的结果,而且可以方便地进行特征的扩充或删减,而不会对整个系统的结构产生影响,参见图1。但是其缺点也是明显的,不同特征之间完全独立,忽视了特征之间的关联关系所带来的作用和影响。解决这个问题需要在较低层级,如参数级、特征级等层面上,展开更深入的融合研究。

(3) 融合的策略问题 数据融合需要通过一定的策略与数学方法将特征和数据加以综合,这些策略和方法的研究也是目前受到普遍关注的问题。关于数据融合与决策目前有多种算法:决策树、线性判别函数、线性分类器、贝叶斯决策、K近邻(KNN)、支持向量机(SVM)等。

参考文献

1. Ross A, Jain A, and Qian J. Information Fu-

sion in Biometrics. In: Proc. of 3rd Int'l Conference on Audio- and Video-Based Person Authentication, Sweden, June 6-8, 2001

2. Frischholz R, Dieckmann U. BioID: A Multimodal Biometric Identification System. Computer, 2000, 33(2): 64~68

3. Chibelushi C, Deravi F, and Mason J. A Review of Speech-Based Biomodal Recognition. IEEE Trans. Multimedia, 2002, 4(2): 23~27

(吴志勇 蔡莲红)

duotai leixing

多态类型 (polymorphic type) 多态类型及其演算研究始于J. Y. Girard的关于二阶直觉主义逻辑的类型系统和J. Reynolds的多态程序的类型系统。多态类型的语法理论有高阶和二阶之分。高阶多态类型分两层次:先定义种属和构造子,后定义类型。一个项的种属、类型依赖于所含变元的种属、类型。因此种属、类型的说明都涉及它们与变元的联系。归纳地定义有效种属: $T, Type$ 都为种属,若 K_1, K_2 为种属,则 $K_1 \times K_2$ (序偶), $K_1 \rightarrow K_2$ (抽象)也都为种属。种属环境为构造子变元连同其规定的种属组成的有限集 $\Delta = [x_1: K_1, \dots, x_n: K_n]$,其中基本表示“ $x_i: K_i$ ”可读成变元 x_i 有种属 K_i 。有效构造子是指构造子变元被环境所说明者。

种属规则(规则陈述如何赋种属):若 Δ 为种属环境。

(1) 若 $x: K \in \Delta$,则 $\Delta \vdash x: K$;(可读成:环境 Δ 中对变元 x 赋种属 K ,以下类同)

(2) $*$: T ;

(3) 若 $\Delta \vdash C_i: K_i (i=1, 2)$,则 $\Delta \vdash \langle C_1, C_2 \rangle: K_1 \times K_2$ (序偶);

(4) 若 $\Delta, x: K_1 \vdash C: K_2$,则 $\Delta \vdash \lambda x: K_1. C: K_1 \rightarrow K_2$ (抽象);

(5) 若 $\Delta \vdash F: K_1 \rightarrow K_2, \Delta \vdash C: K_1$,则 $\Delta \vdash FC: K_2$ (作用)。

类型定义为种属是 $Type$ 的构造子(Δ 为种属环境):

(1) 若 $X: Type \in \Delta$,则 $\Delta \vdash X: Type$;

(2) $1: Type$;

(3) 若 $\Delta \vdash A: Type, \Delta \vdash B: Type$,则 $\Delta \vdash A \times B: Type$ (序偶), $\Delta \vdash A \rightarrow B: Type$ (抽象);

(4) 若 $\Delta, x: K \vdash A: Type$,则 $\Delta \vdash \forall x: K. A: Type$ 。

类型环境是自由变元连同规定的类型组成的集合 $\Gamma = [X_1: A_1, \dots, X_n: A_n]$ 。若 $\Delta \vdash A_1, \dots, A_n$, 则 Γ 称为在 Δ 之下良定义。

类型的规则(陈述如何在 Γ 中赋类型): Γ 为 Δ 之下良定义。

- (1) 若 $x: A \in \Gamma$, 则 $\Gamma \vdash_\Delta x: A$;
- (2) $\Gamma \vdash_\Delta *: 1$;
- (3) 若 $\Gamma \vdash_\Delta a_i: A_i (i=1, 2)$, 则 $\Gamma \vdash_\Delta \langle a_1, a_2 \rangle: A_1 \times A_2$;
- (4) 若 $\Gamma \vdash c: A_1 \times A_2$, 则 $\Gamma \vdash \Pi_i c: A_i (i=1, 2)$ (分量);
- (5) 若 $\Gamma, x: A \vdash b: B$, 则 $\Gamma \vdash_\Delta \lambda x: A. b: A \rightarrow B$;
- (6) 若 $\Gamma \vdash_\Delta f: A \rightarrow B, \Gamma \vdash_\Delta a: A$, 则 $\Gamma \vdash_\Delta fa: B$;
- (7) 若 $\Gamma \vdash_{\Delta x: K} c: C, x$ 不出现在 Γ 中类型, 则 $\Gamma \vdash_\Delta \Lambda x: K. c: \forall x: K. C$;
- (8) 若 $\Gamma \vdash_\Delta F: \forall x: K. C, \Gamma \vdash D: K$, 则 $\Gamma \vdash_\Delta FD: C[D/X]$ 。

归约等式定义为:

- (1) $(\Lambda X: K. F)C \stackrel{\beta^2}{=} F[C/X]$;
- (2) $\Lambda X: K. FX \stackrel{\eta^2}{=} F$;
- (3) $(\lambda x: A. f)a \stackrel{\beta}{=} F[a/x]$;
- (4) $\lambda x: A. fx \stackrel{\eta}{=} f$;
- (5) $\Pi_1 \langle t_1, t_2 \rangle \stackrel{\Pi_1}{=} t_1$;
- (6) $\Pi_2 \langle t_1, t_2 \rangle \stackrel{\Pi_2}{=} t_2$;
- (7) $\langle \Pi_1 t_1, \Pi_2 t_2 \rangle \stackrel{\sigma}{=} t$;
- (8) $u \stackrel{!}{=} *$ 。

其中 u 为 T 或 $1, t$ 为类型 $A_1 \times A_2$, 或为种属 $K_1 \times K_2$, 以上为高阶演算。二阶演算只是省略种属直接考虑类型 $Type$ 。多态类型演算具有丘奇-罗瑟性质和强典范性质。基于多态类型演算可定义二阶直觉主义逻辑:

$\top \rightarrow P \rightarrow Q$	定义为 $P \rightarrow Q$
	定义为 $\forall X: Type. X \rightarrow X$
\perp	定义为 $\forall X: Type. X$
$P \wedge Q$	定义为 $\forall X: Type. (P \rightarrow Q \rightarrow X) \rightarrow X$
$P \vee Q$	定义为 $\forall X: Type. (P \rightarrow X) \rightarrow (Q \rightarrow X)$
$\neg P$	定义为 $P \rightarrow \perp$

$\exists x: A. P[X]$ 定义为 $\forall X: Type. (\forall x: A(P[X] \rightarrow X)) \rightarrow X$

参考文献

1. Girard J. Y. Interprétation Fonctionnelle et Élimination des Coupures Dans L'arithmétique D'ordre Supérieure. Ph. D Thesis, Université Paris VII, 1992
 2. Reynolds J. Polymorphism is not Set-Theoretic. In: Kaha G. et al. Semantics of Data Types. 145 ~ 156. Springer-Verlag, 1984
 3. Reynolds J, Plotkin G. On Functors Expressible in the Polymorphic Typed Lambda Calculus. LFCS Report Series 53, University of Edinburgh, May, 1988
- (陆汝占)

duoxiangshi kongjian guiyue

多项式空间归约 (polynomial space reduction) 一种特殊的、归约函数在多项式空间可计算的复杂性归约。

设 L_1, L_2 是 Σ 上的两个语言, 若存在函数 $S: N \rightarrow N$, 及 S 空间可计算函数 $f: \Sigma^* \rightarrow \Sigma^*$, 使得

- (1) 对任何 $x \in \Sigma^*, x \in L_1$ 当且仅当 $f(x) \in L_2$;
- (2) 存在正整数 C , 使对任何 $x \in \Sigma^*$ 有 $S(|f(x)|) \leq CS(|x|)$, 则称 L_1 可 S -空间归约到 L_2 , 记为 $L_1 \leq_S L_2$ 。特别当限制 S 为多项式函数时, 则称 L_1 可多项式空间归约到 L_2 。空间归约中较为重要的一种是限制 S 为对数函数 \log , 称之为对数空间归约, 可对复杂性类进行更“细”的划分, 特别是研究 P 和 $N \log$ 等复杂性类, 其时间资源不超过确定的多项式时间时, 多项式时间归约则无法对其进行分类。

参考文献

- Balázár J L, Díaz J, Gabarró J. Structural Complexity I. Berlin: Springer-Verlag, 1988
- (马绍汉 李大兴)

duoxiangshi puxi

多项式谱系 (polynomial hierarchy) 递归论中克林算术谱系的多项式变形。很多似乎不在 NP 类中的计算问题属于多项式谱系的某一层。多项式谱系的基本思想是 R. Karp 于 1972 年提出的, A. Meyer 和 L. Stockmeyer 在 1973 年给出了多项式谱系的严格形式化定义。

基于多项式时间图灵归约和多项式时间非确定图灵归约的概念, 可建立 P 和 NP 类关于任何语言 L

的相对化定义,它们分别记为 $P(L)$ 和 $NP(L)$,有

$$P(L) = \{L' \subseteq \Sigma^* \mid L' \leq_r^P L\}$$

$$NP(L) = \{L' \subseteq \Sigma^* \mid L' \leq_r^{NP} L\}$$

这种对 P 和 NP 类关于语言的相对化概念,可自然地推广到任何语言类 \mathcal{C} 上:

$$P(\mathcal{C}) = \bigcup_{L \in \mathcal{C}} P(L), \quad NP(\mathcal{C}) = \bigcup_{L \in \mathcal{C}} NP(L)$$

基于这种定义,可将 P 和 NP 视为语言类上的一种算子,且有 $\mathcal{C} \subseteq P(\mathcal{C}) \subseteq NP(\mathcal{C})$, $P(P) = P$, $NP(P) = NP$,从自语言类 P 开始,将算子 NP 重复地作用在其上,便产生一个语言类的无穷递增序列:

$$P, NP, NP(NP), NP(NP(NP)), \dots$$

它们依次记为 $\Sigma_0^P, \Sigma_1^P, \Sigma_2^P, \Sigma_3^P, \dots$, 也即

$$\Sigma_0^P = P, \quad \Sigma_{k+1}^P = NP(\Sigma_k^P), \quad k \geq 0$$

另外,还可定义两类与 Σ_k^P 相关的复杂性类 Π_k^P 和 Δ_k^P :

$$\Pi_k^P = C_0 - \Sigma_k^P = \{L \subseteq \Sigma^* \mid \bar{L} \in \Sigma_k^P\}$$

$$\Delta_0^P = P, \quad \Delta_{k+1}^P = P(\Sigma_k^P), \quad k \geq 0$$

这三种复杂性类有下述基本关系:

$$\Delta_k^P \subseteq \Sigma_k^P \cap \Pi_k^P, \quad \Sigma_k^P \cup \Pi_k^P \subseteq \Delta_{k+1}^P$$

由此可见

$$\bigcup_{k \geq 0} \Sigma_k^P = \bigcup_{k \geq 0} \Pi_k^P = \bigcup_{k \geq 0} \Delta_k^P$$

由 Σ_k^P, Π_k^P 及 Δ_k^P ($k \geq 0$) 所描述的层次结构记为 PH , 并称 PH 为多项式谱系。

多项式谱系也可如同算术谱系那样,用交替量词的形式来表示。两者之间的区别仅仅是存在量词 $\exists y$ 代之以多项式存在量词 $\exists^P y$; 全称量词 $\forall y$ 代之以多项式规模全称量词 $\forall^P y$; 递归集(语言)代之以多项式时间可计算语言。这就是 C. Wrathall 定理: 对于所有 $k \geq 0$

(1) $L \in \Sigma_k^P$ 当且仅当存在 $L' \in P$, 使得 $x \in L$ 当且仅当 $\exists^P y_1 \forall^P y_2 \dots Q_k y_k \langle x, y_1, \dots, y_k \rangle \in L'$ 。其中当 k 为偶数时, Q_k 为 $\forall^P y_k$; 当 k 为奇数时, Q_k 为 $\exists^P y_k$;

(2) $L \in \Pi_k^P$ 当且仅当存在 $L' \in P$, 使得 $x \in L$ 当且仅当 $\forall^P y_1 \exists^P y_2 \dots Q' y_k \langle x, y_1, y_2, \dots, y_k \rangle \in L'$ 。其中当 k 为偶数时 $Q' y_k$ 为 $\exists^P y_k$; 当 k 为奇数时, $Q' y_k$ 为 $\forall^P y_k$ 。

在 Wrathall 定理中的 $\exists^P y$ 意指存在多项式 P , 对于满足 $|y| \leq P(|x|)$ 的某些 $y \in \Sigma^*$; $\forall^P y$ 意指存在多项式 P , 对于满足 $|y| \leq P(|x|)$ 的所有 $y \in \Sigma^*$ 。

多项式谱系的这种表述形式,对于分析计算问

题所处的层次常常更为方便。

对于任意 $k \geq 0$, 令 k -QBF 表示所有以存在量词开始,至多有 k 个交替量词的可满足量化布尔表达式构成的集合; 令 k -QBF 表示所有以全称量词开始,至多有 k 个交替量词的可满足量化布尔表达式构成的集合; 令 QBF 表示所有可满足的量化布尔表达式的集合。由 Wrathall 定理知,

$$k\text{-QBF} \in \Sigma_k^P, \quad k\text{-QBF} \in \Pi_k^P$$

并由 $0\text{-QBF} = P$, $1\text{-QBF} = SAT$, 容易看出, k -QBF 是 Σ_k^P - m -完全的; k -QBF 是 Π_k^P - m -完全的。考虑到 QBF 是 PSPACE- m -完全的这一事实,有

$$NP \subseteq PH \subseteq PSPACE$$

多项式谱系与其他复杂性类,以及多项式谱系自身的各种复杂性类之间的包含关系,究竟是真包含关系还是相等关系,尚待研究。

多项式谱系中一个有趣而重要的未解决问题是其层次是否塌陷。亦即是否存在 $k \geq 0$, 使 $\Sigma_k^P = \Sigma_{k+1}^P$ 。不难证明,若 $\Sigma_k^P = \Sigma_{k+1}^P$, 则 $\Sigma_k^P = \Pi_k^P = \Sigma_{k+1}^P = \Pi_{k+1}^P = \dots = \Sigma_{k+j}^P = \Pi_{k+j}^P$, 对于任何 $j \geq 0$ 成立。从而推出 $\Sigma_k^P = PH$, 亦即多项式谱系塌陷在第 k 层。

由于许多包含在多项式谱系中的复杂性类,都已证明它们属于谱系中的较低层,例如有界误差概率多项式时间类 $BPP \subseteq \Sigma_2 \cap \Pi_2$, 因此,许多人认为多项式谱系很可能塌陷在较低层,但未能给出证明。多项式谱系的塌陷问题常与复杂性类的其他问题存在着某些联系,彼此相互影响。

参考文献

1. Garey M R, Johnson D S. Computers and Intractability, A Guide to the Theory of NP-Completeness. San Francisco: W. H. Freeman and Company, 1979
2. Balc  zar J L, D  az J, Gabarr   J. Structural Complexity I. Berlin: Springer-Verlag, 1988

(马绍汉 李大兴)

duoxiangshi shijian guiyue

多项式时间归约 (polynomial time reduction) 一种常用的、归约函数在多项式时间可计算的复杂性归约。S. Cook 于 1971 年利用多项式时间图灵归约,定义了 NP 类中的“最困难”问题。并证明了判别布尔表达式的可满足性问题(SAT),是这类问题的第一个问题。

假设所考虑的问题都已编码成字母表 Σ 上的

语言(实例的集合)。设 L_1, L_2 是 Σ 上两个语言,若存在以 L_2 为 oracle 集的多项式时间图灵机 M , 其接受的语言为 L_1 , 则称 L_1 多项式时间图灵归约到 L_2 , 记为 $L_1 \leq_p^T L_2$ 。这时, 对 x 是否属于 L_1 的判别可转化为至多 $|x|$ 的多项式个元素是否属于 L_2 的判别, 因此, $L_2 \in P$ 便导致 $L_1 \in P$ 。从这种相对的意义上来讲, L_1 的计算不比 L_2 困难。

\leq_p^T 可以是定义在任何语言类 \mathcal{C} 上的一种二元前序关系, 如果存在 $L \in \mathcal{C}$, 对于任何 $L' \in \mathcal{C}$, 都有 $L' \leq_p^T L$, 则 L 就是 \mathcal{C} 中(在多项式时间图灵归约下)“最困难”的, 称其为 \mathcal{C} -T-完全的。多项式时间图灵归约又称为库克归约。由多项式时间图灵归约的定义, 很自然地可产生另一种重要的多项式时间归约, 即多项式时间非确定图灵归约。多项式时间图灵归约与多项式时间非确定图灵归约的区别仅在于前者使用的是多项式时间确定型 oracle 机器, 后者使用的是多项式时间非确定型 oracle 机器。

R. Karp 于 1972 年利用多项式时间多一归约来刻画 NP 类中的“最困难”问题类。同时, R. Karp 给出了 21 个属于这类问题的实例, 它们涉及到逻辑、图论及组合优化等学科中的经典计算问题。对于 Σ 上的两个语言 L_1, L_2 , 若存在多项式时间可计算函数 $f: \Sigma^* \rightarrow \Sigma^*$, 使得对任何 $x \in \Sigma^*$, $x \in L_1$ 当且仅当 $f(x) \in L_2$, 则称 L_1 多项式时间多一归约到 L_2 , 记为 $L_1 \leq_m^p L_2$ 。这时, $x \in L_1$ 的判别可以通过计算 $f(x)$, 转化成 $f(x) \in L_2$ 的判别。因此, $L_1 \leq_m^p L_2$ 更直观地理解为 L_1 的计算不比 L_2 的计算困难。同 \leq_p^T 类似讨论, \leq_m^p 也可定义在任何语言类 \mathcal{C} 上, 若存在 $L \in \mathcal{C}$, 使对于任何 $L' \in \mathcal{C}$, 都有 $L' \leq_m^p L$, 则称 L 为 \mathcal{C} -m-完全的。多项式时间多一归约又称为卡普归约。

递归论中的其他归约都可通过多项式变形成为一种多项式时间归约。上述介绍的几种归约关系已成为计算复杂性理论的重要工具。

参考文献

1. Garey M R, Johnson D S. Computers and Intractability, A Guide to the Theory of NP-Completeness. San Francisco: W. H. Freeman and Company, 1979

2. Balczár J L, Díaz J, Gabarró J. Structural Complexity I. Berlin: Springer-Verlag, 1988

(马绍汉 李大兴)

duoxin pian mokuai

多芯片模块 (multichip module) 将多个裸露的集成电路芯片直接安装与互连在一块多层高密度的基板上, 再用管壳密封制成的一种多功能微电子器件。它具有组装密度高、电气性能好、延迟时间短、体积和功耗小等特点。多芯片模块由芯片、多层基板和封装外壳三部分组成, 其基本技术包括裸芯片的微焊接技术、多层基板和封装技术等。

用于多芯片模块的微焊接技术主要有载带自动焊(TAB)和倒装焊(FC)两种。载带自动焊是集成电路内、外引线的一种自动群焊法, 它先把载带(镀锡或镀金的铜箔、粘接剂塑料膜制成的具有引线框的柔性印制电路板)用热脉冲焊或超声热压焊焊到芯片焊区的镀金凸台上(内引线焊接), 然后冲剪下此载带, 并将其迅速焊到基板的焊盘上(外引线焊接), 整个焊接过程自动完成。倒装焊为多芯片模块的关键技术之一。它是将芯片焊区直接与基板上的相应焊区对准后进行焊接, 因而互连线最短, 占用基板面积最小。但工艺难度较大, 散热性能差。

用于多芯片模块的基板材料主要有陶瓷基板(Al_2O_3 , AlN , BeO 等)以及硅基板、低温共烧玻璃陶瓷基板(LTCC)、金属基板和金属芯基板等。所用介质材料有二氧化硅、聚酰亚胺等, 其热膨胀系数应与硅的热膨胀系数接近, 并应具有高的导热系数、低的介电常数和介质损耗系数等。

多芯片模块按其结构与工艺特点可分为下列几类: 采用高密度多层印制电路板的 MCM-L 型; 以硅为基板, 二氧化硅为介质层, 铝或铜为导体的 MCM-Si 型; 采用硅、陶瓷或金属为基板, 聚酰亚胺为介质层, 铝或铜为导体的 MCM-D 型。MCM-L 采用的是厚膜技术, 布线较长, 布线板面积也大, 限制了速度的提高。MCM-Si 型采用硅基板, 导热性能好, 且与芯片材料的热膨胀系数相同, 因而便于热设计。MCM-D 的布线电容比 MCM-Si 小, 有利于实现高速化, 但其导热性能不如 MCM-Si 型。MCM-Si 及 MCM-D 型因其布线工艺采用了光刻技术, 有利于提高集成度和高速化, 将成为多芯片模块的主要类型。

多芯片模块已成功地应用于大型计算机和巨型计算机中, 美国 IBM 公司曾在 3081 型大型计算机上采用此技术, 把 118 个裸集成电路芯片安装互连在 30 层的陶瓷基板上, 组成导热模块(TCM)。为了适应日益高速化的要求, 目前正以 MCM-D 为中心, 开展布线板设计、结构优化设计和改进介质材料性能等研究。随着多芯片模块性能的不断改进和成本

的降低,它将被广泛应用于工作站、微型计算机、通信、医疗电子仪器 and 汽车电子仪器等领域。

(赵悼受)

duozhi luoji

多值逻辑 (multiple valued logic) 变元的取值多于两种的逻辑。在通常的逻辑中,命题变元的取值限于真(T或1),假(F或0)两种,称为2值逻辑。例如在“0”,“1”两值之外,增加“u”,以表示非真非假,就得到3值逻辑。在其中选取“ \neg ”与“ \rightarrow ”作为原始联结词,并规定它们的赋值如下:

A	$\neg A$	$A \rightarrow B$	0	1	u
0	1	0	1	1	1
1	0	1	0	1	u
u	u	u	u	1	1

其他命题联结词($\vee, \wedge, \leftrightarrow$)则分别定义为:

$$A \vee B = (A \rightarrow B) \rightarrow B$$

$$A \wedge B = \neg(\neg A \vee \neg B)$$

$$A \leftrightarrow B = (A \rightarrow B) \wedge (B \rightarrow A)$$

根据上述定义,可以得出这些联结词的赋值表:

$A \vee B$	0	1	u	$A \wedge B$	0	1	u	$A \leftrightarrow B$	0	1	u
0	0	1	u	0	0	0	0	0	1	0	u
1	1	1	1	1	0	1	u	1	0	1	u
u	u	1	u	u	0	u	u	u	u	u	1

如果对u作不同的解释,即改变 \neg 与 \rightarrow 的赋值规定,就能得到不同的3值逻辑。在上述3值逻辑中,每一个恒真命题都是2值逻辑中的恒真命题。但反之不然,例如排中律 $\neg A \vee A$ 就是最重要的例外。多值逻辑取值的个数并不限于3,用完全类似的方法,可以构造出n值、 N_0 值、在 $[0,1]$ 上取值或在其他适当集合中取值的逻辑。由于多值逻辑演算对于不完善信息提供了可能性推理的工具,故可用于计算机科学、信号系统的设计及量子力学的计算等方面。

设S是一个n元集合,定义在S上而函数值仍属于S的函数称为n值逻辑函数。当 $n=2$ 时便是2值逻辑函数即布尔函数。当S是无限集时,便称为无限值逻辑函数。一组n值逻辑函数称为完备系或生成系,如果由它们通过迭合运算可生出所有的n值逻辑函数,即任意n值逻辑函数可由它们表示,若一个函数可生出所有的n值逻辑函数,则此函数便

称为谢弗函数,例如, $\max(x, y) + 1 \pmod n$ 便是一个谢弗函数。关于完备系的判定,有著名的完备性定理:一个函数系是完备的必要而且只要该函数系中包含非线性函数、非单调函数、非自对偶函数、非保 $E(<S)$ 函数、非保直接分划函数、非保中心分划函数和非保正则分划函数。每一个函数实际上确定了S内的一个代数运算,一个完备系便确定了一个代数系统。常用的代数系统有波斯特代数: $\max(x, y), \min(x, y), x + 1 \pmod n$ 和模代数: $x + y, x \cdot y$,这里S是一个域,相加函数和相乘函数分别是它的加法运算和乘法运算。

多值逻辑函数与多值逻辑网络密切相关,生成系中的每一个函数就是网络中的一个基本元件,不同的生成系就有不同的基本元件。多值逻辑网络的最优设计问题归结为求出多值逻辑函数的最简表达式。因此多值逻辑函数的代数理论是多值数字系统设计和分析的有力数学工具。多值逻辑函数理论广泛用于计算机科学,例如单向函数可作为计算机密码学中的密钥函数。

一般逻辑电路是两个稳定的状态,而多值逻辑电路至少有三个稳定的状态。多值逻辑电路可分为三大类:①电流型电路,以电流值的大小表示逻辑值,典型的技术有集成注入逻辑(I²L),射极耦合逻辑(ECL)及电流型金属-氧化物-半导体电路(MOS);②电压型电路,以电压值的大小表示逻辑值,典型的技术有多元逻辑电路(NMOS, CMOS及DYL);③电荷型电路,以电荷量的大小表示逻辑值,典型的技术有电荷耦合器件(CCD)。

多值逻辑电路与2值逻辑电路相比具有较强的逻辑功能,同时能较大地提高集成电路的密度,降低集成电路连接的复杂度,容易解决集成片引线限制问题。因此,多值逻辑电路为超大规模集成电路的发展开辟了新的途径。

参考文献

1. Rine D C. Computer Science and Multiple-Valued Logic: Theory and Applications. Revised Edition. Amsterdam: Elsevier Science Publishers, 1984
2. 罗铸楷, 胡谋, 陈廷槐. 多值逻辑的理论及应用. 北京: 科学出版社, 1992
3. 谷超豪主编. 数学词典. 上海: 上海辞书出版社, 1992

(罗铸楷 胡谋)

E

erjinzhi suanshu yunsuan

二进制算术运算 (binary arithmetic operation)

对两个二进制数所进行的加法、减法、乘法和除法运算。当操作数以定点形式表示时,称为二进制定点加、减、乘、除法运算;当操作数以浮点形式表示时,称为二进制浮点加、减、乘、除法运算。

参加运算的操作数可以用原码、补码或反码表示。原码运算的操作数一般以原码形式存放在存储器或寄存器中,运算结果也以原码表示。同样,补码(或反码)运算的操作数和运算结果一般用补码(或反码)表示,操作数以补码(或反码)形式存放在存储器中。但由于原码加、减法运算比较复杂,因此在某些存储原码的计算机中,当执行加减法运算时,先将操作数转换成补码,然后运算,最后将以补码表示的运算结果转换成原码,再存储起来。直接以原码参与加减法运算的计算机很少见。

定点运算

定点加减法运算

原码加减法运算 加法运算规则:两数同号(符号位相同),执行加法(绝对值相加);两数异号(符号位不同),执行减法(绝对值相减),并将绝对值大的数作为被减数。运算结果为绝对值。符号位单独处理。

减法运算规则:两数同号,执行减法(绝对值相减),并将绝对值大的数作为被减数;两数异号,执行加法(绝对值相加)。运算结果为绝对值。符号

位单独处理。

补码加减法运算 补码运算规则如下:

$$[X + Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}} \quad (\text{mod } 2)$$

$$[X - Y]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}} \quad (\text{mod } 2)$$

两个补码数相加,符号位参加运算,两数和的补码等于两数补码之和。

两个补码数相减,可以取减数的负数,进行补码加法运算,结果即为两数之差的补码。

反码加减法运算 反码运算规则如下(假设取每个数为双符号位):

$$[X + Y]_{\text{反}} = [X]_{\text{反}} + [Y]_{\text{反}} \quad (\text{mod } 4 - 2^{-n})$$

$$[X - Y]_{\text{反}} = [X]_{\text{反}} + [-Y]_{\text{反}} \quad (\text{mod } 4 - 2^{-n})$$

这里的反码以 $(4 - 2^{-n})$ 为模,当第一符号位产生进位时,第二符号位有溢出,必须把溢出的进位加到最低位 2^{-n} 上去。

下面举例说明原码、补码和反码3种运算方法与步骤(假设用补码、反码表示的数取双符号位)。

例1 设 $X = 0.1010$, $Y = 0.1011$, $X - Y$ 的运算过程见表1。

例2 设 $X = 0.1011$, $Y = 0.1010$, $X - Y$ 的运算过程见表2。

总之,原码加减法运算比较麻烦;反码加减法运算有时需要将最高位产生的进位信号加到最低位(称为循环加),且零值有两种编码;补码加减法运算简便,且零值只有一种编码,因而得到广泛应用。

表1 原码、补码、反码运算(1)

步 骤	原 码 运 算	补 码 运 算	反 码 运 算
1	比较 X, Y 的大小	$[-Y]_{\text{补}} = 1.0101$	$[-Y]_{\text{反}} = 1.0100$
2	减法运算(大数减小数) $\begin{array}{r} 0.1011 \\ -0.1010 \\ \hline 0.0001 \end{array}$	$\begin{array}{r} [X]_{\text{补}} + [-Y]_{\text{补}} \\ 00.1010 \\ + 11.0101 \\ \hline 11.1111 \text{ (结果)} \end{array}$	$\begin{array}{r} [X]_{\text{反}} + [-Y]_{\text{反}} \\ 00.1010 \\ + 11.0100 \\ \hline 11.1110 \text{ (结果)} \end{array}$
3	确定结果符号为1, 结果为1.0001		

表2 原码、补码、反码运算(2)

步 骤	原码 运算	补码 运算	反码 运算
1	比较 X, Y 的大小	$[-Y]_{\text{补}} = 1.0110$	$[-Y]_{\text{反}} = 1.0101$
2	减法运算 $\begin{array}{r} 0.1011 \\ -0.1010 \\ \hline 0.0001 \end{array}$	$\begin{array}{r} [X]_{\text{补}} + [-Y]_{\text{补}} \\ 00.1011 \\ + 11.0110 \\ \hline 00.0001 \text{ (结果)} \end{array}$	$\begin{array}{r} [X]_{\text{反}} + [-Y]_{\text{反}} \\ 00.1011 \\ + 11.0101 \\ \hline 100.0000 \end{array}$
3	确定结果符号为0, 结果为0.0001		最高位进位加到最低位 $\begin{array}{r} 00.0000 \\ + \quad 1 \\ \hline 00.0001 \text{ (结果)} \end{array}$

定点乘法运算

在20世纪70年代以前,在仅有加减运算的低档小型计算机或70年代末的早期微型计算机中,是用软件(乘法子程序)来实现乘法运算的。

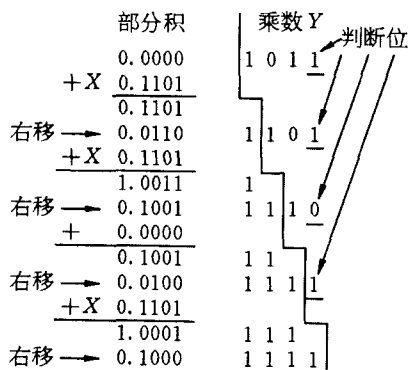
随着集成电路芯片集成度的提高,现在在一般计算机中都有实现乘法运算的硬件,大多与加减法运算共用一个加法器,逐次执行加法与移位操作,分步实现乘法运算。在有些计算机中,为了提高运算速度,设置有专门的乘法器。

常用的定点乘运算方法有原码一位乘法、补码一位乘法、原码两位乘法、补码两位乘法和多位乘法。

原码一位乘法 数值与符号分别处理,两操作数绝对值相乘,符号位相加。两操作数符号相同,乘积为正;符号不同,乘积为负。

执行原码一位乘法时,对应于乘数的每一位(判断位)可得到一项部分积,然后执行1次加法和移位操作,当操作数为 n 位时(内含1位符号位)执行 $n-1$ 次加法和移位操作。

例3 设 $X=0.1101, Y=1.1011$,求 XY 乘积的过程如下:



符号位 $0+1=1$

结果(原码) 1.10001111

补码一位乘法 当参加运算的操作数是补码时,可将数转换成原码,然后按原码一位乘法规则运算。也可采用补码一位乘法,直接对补码进行运算,其运算规则如下:

(1) 被乘数 $X(X=X_0X_1X_2\cdots X_n)$ 和部分积取双符号位,并参与运算。

(2) 乘数 $Y(Y=Y_0Y_1Y_2\cdots Y_n)$ 取单符号位,并在末尾增设附加位 Y_{n+1}, Y_{n+1} 的初始值为0。

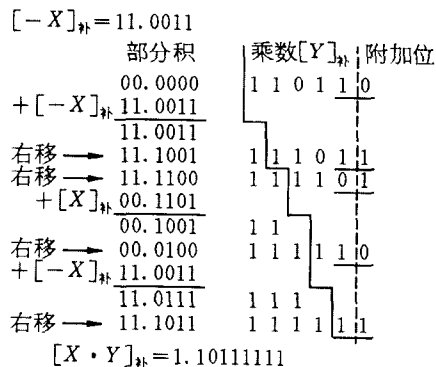
(3) Y_i, Y_{i+1} (第一次为 Y_n, Y_{n+1}) 组成各步运算的乘数判断位,其算法如下:

Y_i	Y_{i+1}	操作
0	0	部分积右移1位
0	1	部分积 $+ [X]_{\text{补}}$, 再右移1位
1	0	部分积 $+ [-X]_{\text{补}}$, 再右移1位
1	1	部分积右移1位

右移时,最高符号位保持不变。

按上述算法执行 $n+1$ 步,但第 $n+1$ 步不移位。

例4 设 $[X]_{\text{补}}=00.1101$ (双符号位), $[Y]_{\text{补}}=1.1011$ (单符号位),求 $[X \cdot Y]_{\text{补}}$ 的过程如下:



这种用比较乘数相邻两位来进行乘法操作的方法

法是由 A. Booth 夫妇首先提出的,所以又叫**布思算法**。

原码两位乘法 为了提高运算速度,在 1 次操作中可同时考虑两位乘数,求得与两位乘数相对应的部分积,其速度比一位乘法提高 1 倍,规则如下:

$Y_i Y_{i+1} = 00$, 相当于 $0 \times X$, 由于是乘两位,部分积右移两位。

$Y_i Y_{i+1} = 01$, 相当于 $1 \times X$, 部分积 $+X$, 然后右移两位。

$Y_i Y_{i+1} = 10$, 相当于 $2 \times X$, 部分积 $+2X$, 然后右移两位。

$Y_i Y_{i+1} = 11$, 相当于 $3 \times X$, 因为 $+3X$ 的实现有困难,所以用 $4X - X$ 来代替,在本步中只执行 $-X$, 用一个欠账触发器记下欠账 C_j , 下一步再补上本步的 $+4X$, 由于本步执行 $-X$ 后部分积要右移 2 位,于是本步的 $+4X$ 操作在下一步只要执行 $+X$ 就可以了。所以原码两位乘法所执行的操作实际上取决于乘数的最低两位 Y_i, Y_{i+1} 和 C_j 的值。

乘法规则如表 3 所示 ($-X$ 用 $[-X]_{补}$ 来代替,被乘数与部分积取 3 个符号位)。

表 3 原码两位乘法

C_j	Y_i	Y_{i+1}	操 作
0	0	0	部分积右移 2 位,置 $C_j = 0$
0	0	1	部分积 $+X$, 然后右移 2 位,置 $C_j = 0$
0	1	0	部分积 $+2X$, 然后右移 2 位,置 $C_j = 0$
0	1	1	部分积 $-X$, 然后右移 2 位,置 $C_j = 1$
1	0	0	部分积 $+X$, 然后右移 2 位,置 $C_j = 0$
1	0	1	部分积 $+2X$, 然后右移 2 位,置 $C_j = 0$
1	1	0	部分积 $-X$, 然后右移 2 位,置 $C_j = 1$
1	1	1	部分积右移 2 位,置 $C_j = 1$

补码两位乘法 将补码一位乘法的布思算法与原码两位乘法结合起来,可推导出补码两位乘法的规则。

多位乘法 可在两位乘法的基础上实现多位乘法,或采用阵列乘法器进一步提高运算速度。

定点小数除法运算 根据操作数表示方式的不同,可分为原码除法和补码除法。原码一位除法具体实现时又可采用恢复余数法或加减交替法。为了

提高运算速度,还可采用跳 0 跳 1 法和迭代法等。

除法运算与乘法运算相似,将 n 位除法操作转换成若干次加減及左移操作,可用硬件或软件实现。

原码一位除法: 数值部分相除,符号位相加。现将恢复余数法与加减交替法的运算规则叙述如下:

恢复余数法 被除数减去除数,如果够减(余数为正或 0),为溢出;如果不够减(余数为负),商 0,并加上除数(恢复余数),被除数左移一位。以后遵循下列规则操作: 余数减去除数,如果够减(余数为正或 0),商 1,余数左移 1 位;如果不够减(余数为负),商 0,并加上除数(恢复余数),然后余数左移 1 位。重复执行,直到商满足精度要求为止。当操作数的数值部分为 n 位时,一般重复执行 n 次。

加减交替法 (不恢复余数) 被除数减去除数,如果够减(余数为正或 0),为溢出;如果不够减(余数为负),商 0,余数左移 1 位,然后遵循下列规则继续操作: 如果上次余数为正或 0,余数减去除数,得新余数;如果上次余数为负,余数加除数,得新余数。如果新余数为正或 0,商 1;新余数为负,商 0。然后将余数左移 1 位。重复执行,直到商满足精度要求为止。

浮点运算

浮点数比定点数的表示范围宽,有效精度高,更适合于科学计算和工程计算。

浮点运算可分成规格化和非规格化运算两种,规格化浮点运算的操作数与运算结果都用规格化浮点数表示。其优点是尾数具有较长的有效位数。除了特殊说明以外,一般浮点运算都为规格化运算。

浮点数由阶码和尾数两部分组成。尾数部分的运算与定点数运算方法相似。在不少计算机中,阶码用移码表示,尾数用补码表示,下面的讨论也以此作为依据。

浮点加减法运算 运算步骤如下:

(1) 对阶。两个规格化浮点数的阶码可能不相等,需要将其阶码统一后才能对尾数进行加减法运算。对阶时首先求两数的阶差,然后将阶值小的数的尾数按阶差右移,使两数具有相同的阶码值。

(2) 尾数进行加减法运算。同定点数运算。

(3) 结果规格化。尾数运算结果可能是不规格化的。有两种不规格化: 第一种是尾数的最高位与符号位相同(即尾数的绝对值小于二分之一); 第二种是尾数的两个符号位不相同(即尾数溢出,超出了定点小数所能表示的范围)。对于第一种不规格化的运算结果,应将尾数向左移位,每左移一位,阶

码减 1,直到尾数的最高位与符号位不同为止。这种操作称为向左规格化,简称左规。如果阶码减至小于机器所能表示的数,称为下溢。此时运算结果用机器 0 表示,即阶码和尾数全为 0。对于第二种不规格化的运算结果,应将尾数右移一位,同时阶码加 1。这种操作称为向右规格化,简称右规。如果阶码加 1 后溢出,即运算结果大于机器所能表示的数,称为上溢。

(4) 舍入。在执行对阶或右规时,有可能使尾数的低位移掉,影响数据的精度,为了将移掉的高位保存起来,应采用适当的方法进行舍入。常用的舍入法为 0 舍 1 入法(类似于十进制的 4 舍 5 入法),如果移掉的最高位为 1,则在尾数的末位加 1;如果移掉的最高位为 0,则舍去移掉的数值。

在尾数末位加 1 后,有可能使尾数溢出,如果发生这一情况,需右规,阶码加 1,并再次判断阶码是否溢出。

浮点乘法运算 运算规则如下:

(1) 阶码相加,如阶码相加后溢出,则表示运算结果溢出。

(2) 尾数相乘,同定点小数乘法。

(3) 向左规格化,并检查下溢。

(4) 舍入,并检查上溢。

浮点除法运算 运算规则如下:

(1) 阶码相减,判溢出(同浮点乘法运算)。检查除数是否为 0,若是,即为溢出。检查被除数是否为 0,若是,结果(商)为 0。

(2) 尾数相除,同定点小数除法,但此处不判溢出。

(3) 规格化,如果右规,阶码加 1,检查上溢。

(4) 舍入,同浮点乘法运算。

二-十进制运算

某些用于数据处理的计算机可直接对二-十进

制编码的数进行运算,用 4 位二进制码来表示 1 位十进制数。例如十进制数 95 的二-十进制编码为 10010101,前 4 位表示 9,后 4 位表示 5,对 1 位二-十进制编码数进行运算时,如结果大于 9,需向高位产生进位信号,并对本位进行修正(+6);如结果不大于 9,则不需要修正。

例 5 ① $8 + 9 = 17$

$$\begin{array}{r} 1000 \\ + 1001 \\ \hline 10001 \\ + 0110 (+6) \\ \hline 10111 \end{array}$$

进位 ↗

② $5 + 6 = 11$

$$\begin{array}{r} 0101 \\ + 0110 \\ \hline 1011 \\ + 0110 (+6) \\ \hline 10001 \end{array}$$

进位 ↗

③ $3 + 5 = 8$

$$\begin{array}{r} 0011 \\ + 0101 \\ \hline 1000 \end{array}$$

参考文献

1. 李勇等. 计算机原理与设计. 修订本. 长沙: 国防科技大学出版社, 1989
2. 王爱英主编. 计算机组成与结构. 第三版. 北京: 清华大学出版社, 2001 (王爱英)

F

fanchuan xuexi

反传学习 (back-propagation learning) 一种前向多层神经网络的学习方法,学习过程由输入的正向传播和误差的反向传播组成。在正向传播过程中,输入信号从输入层经隐层单元逐层处理,并向输出层传播,每一层神经元的状态只影响下一层的状态。当输出存在误差时,则将输出信号与期望输出信号的误差进行反向传播,修改各神经元之间的连接权值,从而减小输出误差。反传学习属于误差修正型学习,所以有时也称为误差最快梯度下降学习算法。

反传网络模型最早是由 P. J. Werbos 在 1974 年提出来的。1985 年 D. Parker 对反传学习算法作了更进一步研究。并行分布处理研究组的 D. E. Rumelhart 等推导出反传学习算法,并且对其功能进行了广泛深入的探讨,系统地解决了多层神经网络中隐层单元连接权值的学习问题。

反传网络的结构由多层前馈非线性可微分单元组成,一般包含有一个输入层,一个输出层,一至多个隐层。同层单元间没有连接,前后层单元间的连接强度用权值表示。图 1 给出了反传神经网络的基本结构。

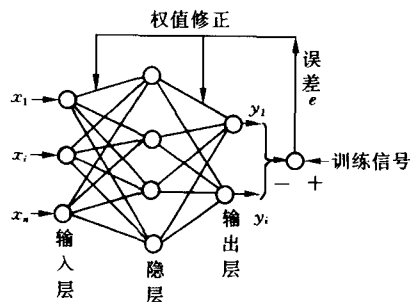


图 1 反传神经网络

反传学习时,通过样本训练,对网络的连接权值进行修改。设输入一训练样本,其输入向量为 X_t , 目标向量为 Y_t , 经过神经网络作用后形成一输出向量 $f(X_t)$, 可用公式

$$E_t = \sum_i |y_{ti} - f_i(X_t)| \quad (1)$$

来计算所修正的误差。式(1)中 y_{ti} 为目标向量 Y_t 的分量, $f_i(X_t)$ 为输出向量的分量, t 为样本编号。全局误差最小有两种形式:一种是针对所有样本或随机选取的某些样本,一种是针对某个样本。根据误差形成的不同,有两种训练方法,前者为样本集中式训练方法,后者为单样本渐进式训练方法。这里以单样本渐进式训练方法为例,介绍反传学习的基本算法。首先,给网络的连接权值随机赋值,取值在 $[-1, \text{eps}, +1]$ 区间。然后取一个样本 X , 计算每一层结点的输入 $\text{net}_j = \sum_i w_{ij} o_i$, 其中 o_i 为前一层结点 i 的输出, w_{ij} 为连接权值。计算输出层的误差 $\delta_j = (T_j - o_j) f'(\text{net}_j)$, 其中 T_j 是相对于样本 X 的期望输出值。如果为隐层结点,其输出误差 $\delta_j = f'(\text{net}_j) \times \sum_m \delta_m w_{mj}$, δ_j 为上一层单元的误差。根据这些误差计算 $\Delta w_{ij} = \mu \delta_j o_i$, 其中 μ 为学习速率, 取值为 $0 < \mu \leq 1$ 。用 Δw_{ij} 修正权值。接着取下一个样本,重复上述步骤,直到输出结点的输出满足要求为止。

前向多层神经网络的反传学习是目前应用最广的学习方法,在模式分类、模糊控制、残缺信息的恢复、函数逼近等领域都得到应用。目前,反传学习仍存在不少问题,网络中隐单元的存在,使梯度下降不能保证全局最小收敛;训练速度太慢;后面样本训练时可能会破坏前面学习的结果;如何设计隐层和隐单元的数目等。这些问题尚待进一步研究。

参考文献

史忠植. 神经计算. 北京: 电子工业出版社, 1993
(史忠植 胡宏)

fanchoulun

范畴论 (category theory) 以抽象数学结构(称为对象)和保结构映射(称为态射)为主要研究对象的新代数学科。

范畴的概念于 1945 年出现在 S. Eilenberg 和 S. MacLane 关于同调代数的工作中。现在,范畴的语言

和基础部分已渗透到数学的很多领域中,并在它们的一些新的发展中起了重要作用。自从 20 世纪 70 年代 ADJ 小组(J. Goguen, J. W. Thatcher, E. G. Wagner 和 J. B. Wright)探讨计算机科学与范畴论的相关性开始,范畴论的一些成果和方法便逐步应用到计算机科学的许多方面,特别是计算机语言学、代数语义学、类型论和形式化技术等方面。这些应用促进了范畴论的发展。可以相信,像集合论一样,范畴论最终也将找到通向初等水平数学的道路。

范畴 一个范畴 \mathcal{C} 由以下数据和公理组成:

数据 1 一个 \mathcal{C} 对象的类 $\text{ob}\mathcal{C}$ 。

数据 2 一个 \mathcal{C} 态射的类 $\text{mor}\mathcal{C}$ 。每个 \mathcal{C} 态射 f 都对应于一个 \mathcal{C} 对象的有序偶 $\langle A, B \rangle$, 记为 $f: A \rightarrow B$ 或 $A \xrightarrow{f} B$, 并称 A 为 f 的论域, B 为 f 的余论域。 \mathcal{C} 态射 f 的论域和余论域分别用 $\text{dom}(f)$ 和 $\text{cod}(f)$ 表示, 并令

$$\mathcal{C}[A, B] = \{f \in \text{mor}\mathcal{C} \mid \text{dom}(f) = A \text{ 且 } \text{cod}(f) = B\} \quad A, B \in \text{ob}\mathcal{C}.$$

数据 3 对每个 \mathcal{C} 对象 A , 都有一个 \mathcal{C} 态射 $\text{id}_A \in \mathcal{C}[A, A]$, 称为 A 的幺态射。

数据 4 若 \mathcal{C} 态射 f 和 g 使 $\text{dom}(g) = \text{cod}(f)$, 则 f 和 g 对应于一个 \mathcal{C} 态射, 用 $g \circ f$ 或 $g f$ 表示, 称为 f 与 g 的合成态射。

公理 1 若 \mathcal{C} 对象 A, B, A' 和 B' 使 $\langle A, B \rangle \neq \langle A', B' \rangle$, 则 $\mathcal{C}[A, B] \cap \mathcal{C}[A', B'] = \emptyset$ 。

公理 2 若 \mathcal{C} 对象 A 和 B 及 \mathcal{C} 态射 f 使 $\text{dom}(f) = A$ 且 $\text{cod}(f) = B$, 则 $f \cdot \text{id}_A = f = \text{id}_B \cdot f$ 。

公理 3 若 \mathcal{C} 态射 f, g 和 h 使 $\text{dom}(g) = \text{cod}(f)$ 且 $\text{dom}(h) = \text{cod}(g)$, 则 $(h \circ g) \circ f = h \circ (g \circ f)$ 。

如果以集合为对象, 函数为态射, 则可构成一个范畴 set , 称为集合范畴。如果以群、环或域为对象, 相应的同态为态射, 则可分别构成群范畴 grp , 环范畴 rng 或域范畴 field 。如果以半序集(或拓扑空间)为对象, 以单调函数(相应的为连续函数)为态射, 则又可构成半序集范畴 poset (相应的为拓扑空间范畴 top)。这类范畴称为具体范畴。

如果把函数式程序设计语言 \mathcal{L}_{FP} 的类型和函数符号分别作为对象和态射, 则可把 \mathcal{L}_{FP} 看做一个范畴。如果把一个逻辑形式系统 FSP 的合式公式和形式证明分别作为对象和态射, 则 FSP 也可看作一个范畴。

如果把范畴 \mathcal{C} 的每个 \mathcal{C} 态射 f 都反向, 即把 f 的论域和余论域对调, 则可获得一个新范畴, 称为 \mathcal{C}

的对偶范畴, 记为 \mathcal{C}^{op} , 其态射用 f^{op} 表示。

如果 \mathcal{C} 对象 A 到每个 \mathcal{C} 对象都恰有一个 \mathcal{C} 态射, 则称 A 为 \mathcal{C} 的初始对象, \mathcal{C}^{op} 的初始对象称为 \mathcal{C} 的终止对象。同集合论中的内射、满射、双射和逆函数相对应的态射分别是单态射、满态射、同构态射和逆态射。在范畴中还可以定义积、等子、回拉和极限, 以及它们的对偶概念余积、余等子、外推和余极限。

范畴的对偶原理 一个命题为真当且仅当它的对偶命题(即命题中的概念都用其对偶概念替换)为真。

范畴 \mathcal{C} 的图和交换图 若图 $D = \langle V, E \rangle$ 使 $V \subseteq \text{ob}\mathcal{C}, E \subseteq \text{mor}\mathcal{C}$, 且当 $e \in E$ 是 D 中一条从顶点 A 到 B 的有向边时, 皆有 $e \in \mathcal{C}[A, B]$, 则称 D 为 \mathcal{C} 的一个图。称 \mathcal{C} 的以下三角形图和四边形图是交换的(图 1), 是指 $g \circ f = h$ 和 $f_2 \circ g_1 = g_2 \circ f_1$:

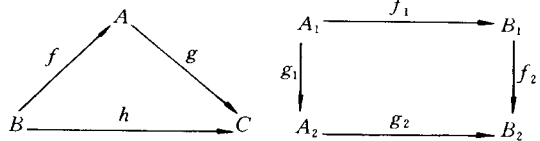


图 1 两图是交换的

如果 \mathcal{C} 的图 D 的每个三角形子图和四边形子图都是交换的, 则称 D 为交换的。

函子 设 \mathcal{C} 和 \mathcal{D} 为两个范畴。如果函数 $F_0: \text{ob}\mathcal{C} \rightarrow \text{ob}\mathcal{D}$ 和 $F_1: \text{mor}\mathcal{C} \rightarrow \text{mor}\mathcal{D}$ 满足:

(1) 若 $f \in \mathcal{C}[A, B]$, 则 $F_1(f) \in \mathcal{D}[F_0(A), F_0(B)]$;

(2) 若 $A \in \text{ob}\mathcal{C}$, 则 $F_1(\text{id}_A) = \text{id}_{F_0(A)}$;

(3) 若 $f \in \mathcal{C}[A, B]$ 且 $g \in \mathcal{C}[B, C]$, 则 $F_1(g \circ f) = F_1(g) \circ F_1(f)$ 。

则称 $F = \langle F_0, F_1 \rangle$ 为一个从 \mathcal{C} 到 \mathcal{D} 的协变函子, 记为 $F: \mathcal{C} \rightarrow \mathcal{D}$ 或 $\mathcal{C} \xrightarrow{F} \mathcal{D}$, 并称 \mathcal{C} 为 F 的定义域, \mathcal{D} 为 F 的值域。

称从 \mathcal{C}^{op} 到 \mathcal{D}^{op} 的协变函子为从 \mathcal{C} 到 \mathcal{D} 的反变函子。

若对任意群 $\langle G, \cdot \rangle$ 和任意群同态 $f: \langle G, \cdot \rangle \rightarrow \langle G', * \rangle$, 皆令 $U(\langle G, \cdot \rangle) = G$, 并定义函数 $U(f): G \rightarrow G'$ 如下:

$$U(f)(x) = f(x), \quad x \in G$$

则可获得一个协变函子, $U: \text{grp} \rightarrow \text{set}$ 。这种从具体范畴到集合范畴的协变函子称为忘却函子, 因为它

是通过“忘掉”具体范畴的结构而得到的。

如果范畴 \mathcal{C} 的对象类 $\text{ob}\mathcal{C}$ 和态射类 $\text{mor}\mathcal{C}$ 都是集合, 则称 \mathcal{C} 为小范畴。以小范畴为对象, 小范畴间的函子为态射, 可构成一个范畴 cat , 称为小范畴的范畴。

自然变换 设 \mathcal{C} 和 \mathcal{D} 为范畴, F 和 G 都是从 \mathcal{C} 到 \mathcal{D} 的协变函子。称 $\eta = (\eta_A)_{A \in \text{ob}\mathcal{C}}$ 为一个从 F 到 G 的自然变换, 并记为 $\eta: F \rightarrow G$ 或 $F \xrightarrow{\eta} G$, 是指对每个 $f \in \mathcal{C}[A, B]$ 皆有图 2 所示的交换图。

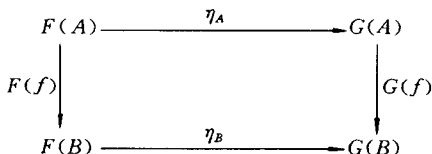


图 2 交换图

若 \mathcal{C} 和 \mathcal{D} 都是小范畴, 则以从 \mathcal{C} 到 \mathcal{D} 的协变函子为对象, 相应的自然变换为态射, 可构成一个范畴 $\text{func}(\mathcal{C}, \mathcal{D})$ 。

参考文献

1. Barr M, Wells C. Category Theory for Computing Science. New York: Prentice Hall, 1990
2. Pierce B C. Category Theory for Computer Scientists. London: The MIT Press, 1991

(王兵山 陈意云)

fanshi

范式 (normal form) 满足一定基本条件的关系模式 (参见模式)。

根据条件的强弱程度, 分别称满足这些条件的关系模式为第一范式 (1NF)、第二范式 (2NF)、第三范式 (3NF)、BC 范式 (BCNF)、第四范式 (4NF) 等。第一范式规定数据库中用以表示实体的属性值必须为不可再分的单个数据。在第一范式中: ①若非键码属性不函数依赖于任一键的真子集时称此模式是第二范式 (参见数据依赖、键码); ②若非键码属性不函数依赖于任一非键码属性集时称为第三范式; ③所有非平凡函数依赖都是对键码的依赖时称为 BC 范式; ④所有非平凡函数和多值依赖都是对键码的依赖时称为第四范式。所以关系数据库实际上是第一范式关系数据库 (参见规范化)。

第一范式的目的在于简化关系模型, 避免复杂的数据结构。其他各级范式都逐级地加强条件使关

系模式表达的概念单一化, 由此消除更多的数据冗余, 从而逐级地加强数据一致性。

参考文献

Ullman J D. Principles of Database Systems. Second Edition. London: Pitman Publishing Ltd., 1987

(楼荣生 朱扬勇)

fanghuoqiang

防火墙 (firewall) 建立在内、外网络边界上的过滤封锁机制。内部网络被认为是安全和可信的, 而外部网络 (通常是 Internet) 被认为是不安全和不可信赖的。防火墙的作用是防止不希望的、未经授权的通信进出被保护的内部网络, 通过边界控制来强化内部网络的安全政策。

防火墙技术可以分为网际协议 (IP) 过滤、线路过滤、应用层代理和状态检测等类型, 目前越来越多的防火墙混合使用这些技术, 以获得最大的安全性和最高的系统性能。防火墙系统通常由过滤路由器和代理服务器组成。过滤路由器是一个多端口的 IP 路由器, 它依据一组规则对每一个到来的 IP 包进行检查, 以判断是否对之进行转发。过滤路由器从包头取得信息 (例如, 协议号、收发报文的 IP 地址和端口号, 连接标志以至另外一些 IP 选项), 对 IP 包进行过滤。

代理服务器是防火墙系统中的服务器进程。它能够代替网络用户完成特定的基于传输控制协议和网际协议 (TCP/IP) 的应用。代理服务器本质上是应用层网关, 即为特定网络应用而连接两个网络的网关。用户就一项基于 TCP/IP 的应用 (比如远程通信 (Telnet) 或者文件传送 (FTP)) 同代理服务器打交道。代理服务器要求用户提供其要访问的远程主机名。当用户答复并提供了正确的用户身份及认证信息后, 代理服务器就连通远程主机, 为两个通信点充当中继。整个过程可以对用户完全透明。用户提供的用户身份及认证信息可用于用户级的认证。最简单的情况是: 它只由用户标识和口令构成。但是, 如果防火墙是通过 Internet 访问的, 就需要使用更强的认证机制, 比如使用一次性口令或挑战-响应系统 (参见鉴别)。

过滤路由器的优点是: ①结构简单, 可以通过硬件实现来获得高效率, 而且受到广泛支持; ②(硬件) 成本低; ③对上层协议和应用透明, 无需修改已有的应用。它的可用性是没有问题的, 对于一般应用来说能够防止大部分安全攻击, 但是对于某些关

键任务的应用却存在很多安全和管理方面的漏洞。从网络安全角度来看其主要缺点是在认证和访问控制方面粒度太粗。认证只能针对主机 IP 地址,无法做到用户级别的身份认证,因此存在假冒 IP 攻击的隐患;访问控制也只能控制到 IP 地址端口一级,不能细化到文件等具体对象。从系统管理角度来看人工负担很重,正确建立包过滤规则比较困难,系统管理员需要不断对所有的地址变化作出相应的反应,而系统管理员并不是总能够立即获得这些地址的变化信息的。在复杂的网络环境下,特别是在拥有多个进行 IP 过滤的防火墙或者路由器的情况下,保证 IP 分组过滤表的正确设置(包括一致性和实时更新等方面)是很困难的。

代理服务器的优点是用户级的身份认证、日志记录和账号管理。其缺点在于要想提供全面的安全保证,就要对每一项服务都建立对应的应用层网关,这就严重地限制了新应用的采纳。日本 NEC 公司提出的 SOCK 5(RFC 1928)作为通用应用的代理服务器,主要由一个运行在防火墙系统上的代理服务器软件包和一个链接到各种网络应用程序的库函数包组成。新的应用只要对原先 SOCKET 接口处作简单修改就可以利用 SOCK 5 提供的安全服务。SOCK 4 只支持基于传输控制协议(TCP)的应用,SOCK 5 已经加入了用户数据报协议(UDP)的支持,并且被因特网工程任务部(IETF)接受为 RFC 1928。这样的结构有利于新应用的挂接,目前 Netscape 的 Navigator 浏览器软件、Microsoft 的 Internet Explorer 等都已经支持 SOCK 5。

防火墙给使用也带来很多不便之处。例如,需要进行多次登录以及其他受约束的机制会影响 Internet 的使用。还有人认为防火墙给人制造一种虚假的安全感,认为有了防火墙便保证了安全,导致在防火墙内部放松安全警惕。根据美国的一项调查统计,严重的网络安全问题大多数是内部犯罪引起的,这是任何基于边界隔离的防范措施都无能为力的。同样,防火墙也不能解决进入防火墙的数据带来的安全问题。如果用户下载一个程序在本地运行,那个程序很可能就包含一段恶意的代码,或泄露敏感信息,或对系统进行破坏。随着 Java、JavaScript 和 ActiveX 控件及其相应浏览器的大量使用,这一问题将变得更加突出和尖锐。人们已经认识到,防火墙不能替代内部网络的安全措施。防火墙作为解决一些机构对于网络边界安全的迫切需求起了相当大的作用,而且在当今 Internet 世界中是有生命力的。但

是它并不是解决所有网络安全问题的灵丹妙药,更不能认为网络安全措施就是建立防火墙,它只是网络安全政策和体系中的一个组成部分,只能解决网络安全的部分问题。

参考文献

1. 胡道元. 计算机网络(高级). 北京: 清华大学出版社, 1999
2. Dan Blacharski. Network Security in a Mixed Environment. IDG Books Worldwide Inc., 1998

(胡道元)

fang xinxi xielou jishu

防信息泄漏技术 (technique of electro-mechanical protection against encission and spurious transmission, TEMPEST) 研究抑制计算机及其外围设备等信息设备的电、磁、声等信号杂散发射的技术。

计算机等电子设备工作时,会向空间发射电、磁、声信号,这些信号一旦被他人接收分析,即可得知计算机所处理的信息内容,从而造成泄密。随着科学技术的发展,有时使用不太复杂的设备也可截获计算机的各种信息。例如,可在 1 km 以外用一种不太复杂的装置接收并复现计算机显示器屏幕的信息。计算机的信息泄漏不只限于显示屏幕,计算机主机、键盘、打印机等都会造成信息泄漏,只是其接收难易程度不同而已。

TEMPEST 一词最初是美国政府一项绝密计划即控制电子设备泄密发射的代号。后来成了研究各种信息设备泄密发射的代名词。防信息泄漏技术主要包括以下三方面的研究内容。

标准及规范研究 标准及规范研究的主要内容是信息泄漏极限值、信号检测方法、设备使用环境要求以及使用环境的评估。美国在 20 世纪 70 年代已制定了一套较完善的 TEMPEST 标准和规范(即 NACSI M5100 系列标准)。随着技术的进步,该系列标准分别在 1974 年和 1984 年进行了更新。90 年代初,随着冷战的结束和国际形势的变化,以及 TEMPEST 技术发展的实际情况,美国对原有的 TEMPEST 标准作了重新修改并再度颁布,新的标准包括 NSTISSAM, NSTISSP, NSTISSI 等系列标准。新标准的最大变化在于给防护设备分级,对使用环境分级,以适应不同用户的需求。北约在 1982 年颁布了自己的 TEMPEST 标准,即 AMSC 系列标准,分别

为 AMSG720B, AMSG788, AMSG784。英国的 TEMPEST 标准为 BTR/01/202。我国于 1999 年颁布了自己的 TEMPEST 标准。

防护及制造技术研究 虽然屏蔽室、干扰器和 TEMPEST 都可以起到防信息泄漏的目的,但只有 TEMPEST 是防止计算机信息泄漏的根本措施。防信息泄漏设计中的一项核心工作是红信号和黑信号的划分,包含机密信息且又未经加密的信号称为红信号,存在红信号的区域称红区。不包含机密信息或经加密处理过的信号称为黑信号,只存在黑信号的区域称黑区。防护信息泄漏的基本措施是抑制红信号的发射和防止红信号在红区和黑区之间传输。为了抑制红信号的发射和传输,常采用两种方法:一种是包容法,主要采用屏蔽、滤波、隔离等技术对设备内部各单元或设备本身进行设计或改造,从而抑制红信号的传输及发射。另一种方法是抑源法,主要是从产品最初设计阶段开始,从电路设计、布线及元器件选择等方面抑制红信号的传输和发射,从而使最终产品满足防信息泄漏要求。传统的防护技术必须采取一定的物理处理措施,设备的改造成本较高。后来有人提出 Soft-Tempest 技术概念,其技术优点在于利用软件来降低视频信号的辐射发射,防护成本低。

检测技术研究 主要包括红信号检测方法、测试系统组成与专用检测设备的研制等内容。

红信号的检测分析是防信息泄漏技术特有的一项内容。常采用特征识别法和相关比较法,通过测量信号脉冲宽度、波形上升及下降时间、脉冲重复周期等参数,从时域和频域分析信号特征,从而最终区分红黑信号。

防信息泄漏检测所需设备主要包括 TEMPEST 测试接收机、各种天线及探头、信号分析仪器及屏蔽室。其中 TEMPEST 测试接收机为防信息泄漏测试专用设备,要求设备有较高的灵敏度,中频带宽较宽且可调节,中频波形因数好,自动化程度高,并具有高性能的前置及后置滤波器以及多种解调及输出方式等。

(於亮)

fangzhen yuyan

仿真语言 (simulation language) 一种面向仿真过程的专用计算机语言。仿真模型的执行最终是通过执行计算机程序来实现的。为使仿真程序得以产生,就需要以适用的仿真语言进行程序设计,以便将实际系统的模型用程序加以描述和执行(参见

计算机仿真)。

早期的仿真都是仿真人员根据模型及实验要求采用通用程序设计语言来设计程序,这就要求仿真人员不但要熟悉仿真的对象,而且要熟悉计算机及程序设计语言,大大限制了计算机技术在仿真中的应用。

为简化仿真模型的准备及编程工作,各种通用的仿真程序包相继产生。它实际上是一种根据某类系统仿真需要而事先编好的程序,仿真人员只需按照其事先设计好的格式来输入有关参数、数据,仿真程序包就可自动构造出仿真模型,编排好仿真计算次序,并可加以执行,输出仿真结果。这样研究人员就可将精力集中在所研究的对象上,而不必花费大量时间用于程序的开发与调试。这对于推广计算机在仿真领域的应用起了极大的促进作用。

仿真程序包是仿真语言的初级形式。尽管仿真程序包具有一定的通用性,但由于是事先设计好的程序,因而模型描述能力、实验控制能力及输入输出能力总是有限的。为适应更为复杂的系统仿真建模的要求以满足各种仿真实验运行的需要,在仿真程序包的基础上产生了仿真语言。

仿真语言与仿真程序包的最大的区别在于,仿真语言有一套完整的描述仿真对象及仿真实验所需要的符号、语句及语法,其相应的编译程序能检测出语法错误及逻辑错误。

仿真语言经历了四个发展阶段:

第一阶段为初级语言阶段(1960 年—1970 年)。代表性的有 MIDAS, CSSL, GPSS, GASP 等,它们提供统一的建模、实验、统计输出的框架。

第二阶段为高级仿真语言阶段(1970 年—1980 年)。代表性的有 CSSL IV, DAREP, ACSL, GPSS IV, SIMSCRIPT II 5, SLAM, SIMULA 等。其特点是具有多种建模观点,仿真过程具有跟踪能力,增加了统计分析功能;特别是,由于计算机技术的发展,仿真语言的宿主机逐步过渡到以小型机为主。

第三阶段为一体化建模与仿真环境软件阶段。代表性的是 TESS,它具有在数据库基础上实现数据存储与检索、脚本仿真、数据收集、数据分析、报告及图形生成、脚本动画、网络模型输入、运行控制、数据管理等功能的集成。

第四阶段为智能化仿真软件环境阶段。它于 20 世纪 80 年代后期问世,由一体化仿真环境、专家系统及智能接口等组成,并具有知识库、模型库、方法库、实验框架库的集成功能。目前,这一阶段的研

究工作还在进行中。

国内外流行的仿真语言有多种,它们分别用于不同的领域。按照系统模型的种类,可分为两大类,一类称为**连续系统仿真语言**(面向常微分方程、偏微分方程、差分方程模型),另一类称为**离散事件系统仿真语言**(面向离散事件系统模型,如排队模型、PERT网模型、PETRI NET模型等)。也有一些语言同时具有上述两种模型仿真的能力,但其实质仍然是为了解决两类模型在仿真时的相互控制和通信技术,从仿真建模方法学的观点来看仍然可按两大类考虑。

下面,从四个方面对常见的仿真语言加以说明。

(1) 表达模型的能力 许多先进的连续系统仿真语言,均为面向问题方式,用多种形式表示常微分方程描述的模型(包括方程组、向量、间断右端函数、传递函数、框图等形式)和差分方程描述的模型(包括方程型及Z变换形式)。至于对偏微分方程(包括抛物型、椭圆型、双曲型、双调谐型)模型的仿真,则趋向于做成专门求解某一类型的专用程序包形式。

离散事件系统仿真语言一般采用事件调度、活动扫描、进程交互、三阶段中的一种或多种策略来描述模型(参见**离散事件系统仿真建模方法学**)。此外,人们还结合具体应用领域(如**计算机集成制造系统**、**柔性制造系统**、计算机系统、网络系统等)发展更为专用的面向问题描述的离散事件系统仿真语言。

(2) 语言结构特性 包括既要便于建立模型和进行实验,又需考虑其翻译的可能性。

在连续系统仿真语言中,大多允许按任意顺序描述模型,即具有所谓“并行”结构。仿真语言具有排序算法,许多连续系统仿真语言中引入了“子模型”的概念,有些还实现了“段”结构,以适应多帧速、实时仿真及多处理机仿真的需要。

离散事件系统仿真语言常借助于流程图及PERT网描述,实现并行结构,并以过程并发的形式实现仿真钟的推进。

结构化递阶建模的思想已在新的语言中越来越多地得到使用,有些语言还实现了模型与实验框架分离描述,从而实现模型的数据驱动。

(3) 语言使用方式 早期的仿真语言大多是非交互式的。近年来,交互式语言已成为发展方向。大多数流行的仿真语言可以交互改变模型参数、初始条件、控制仿真实验过程等(无需重新编译、连

接),但是模型结构的交互改变还只见于少数的仿真语言中。

(4) 输入输出能力 早期的仿真语言其输入一般采用文件方式,而其输出是标准的数据报告。现在,已广泛采用图形、菜单、表格、图标等方式输入模型及参数,以漂亮的图形、动画(实时的、二维的或三维的)输出显示已是普遍的现象。

参考文献

文传源. 系统仿真学科与仿真系统技术. 系统仿真学报, 1992, 4(3) (肖田元)

fangwen kongzhi

访问控制(access control) 确定来访实体是否具有访问的权力以及实施访问权限的管理的过程和技术。被访问的数据,如文件、数据报文、分组数据包、数据帧等,统称客体。能访问或使用客体的活动实体称作主体,如用户以及作为用户代理的进程、作业或任务等。访问控制一般都是基于安全策略和安全模型的。Lampson提出的访问矩阵是表示安全策略的最常用的访问控制安全模型。该矩阵中“列”表示访问者,即主体;“行”表示被访问对象,即客体。访问者对访问对象的权限就存放在矩阵中对应的交叉点上。

为节省存储空间实际系统通常并不直接对矩阵进行操作,而是通过访问控制表或者访问权力表进行操作。**访问控制表(ACL)**按照“行”来存储矩阵。它是在对象服务器上储存着每个对象的授权访问者及其权限的一张表。负责保护访问对象的程序称为引用监控器。它根据访问控制表(ACL)的内容来判断是否授权某个访问者某些访问权限。访问权力表则按照“列”来存储矩阵。每个访问者都储存有访问权力表,该表包含了它能够访问的特定对象和操作权限。引用监视器根据验证访问表提供的权力表和访问者的身份来决定是否授予访问者相应的操作权限。

根据能够控制的访问对象粒度我们可以将访问控制分为粗粒度访问控制、中粒度访问控制和细粒度访问控制。这里并没有严格定义的区分标准,但是人们通常将能够控制到文件甚至记录对象的访问控制称为细粒度访问控制。而只能控制到主机对象的访问控制则称为粗粒度访问控制。

目前,很多计算机系统的安全都是采用访问控制表(ACL)访问控制模型。分布式系统和网络系统也不例外。ACL模型提供安全保密和完整性安全

策略的基础。

源通信参与方是通信发起者和请求者。请求信息包含了对网络资源进行某种操作的请求。ACL服务器通过引用监控器检查源通信方的请求内容并决定是否允许通过。访问对象是网络资源,如文件、设备或者中央处理器(CPU)等。

在集中式系统中访问控制是很容易实现的,因为操作系统控制着所有访问对象并且管理所有进程,所有操作均在主机操作系统管理下进行。在分布式系统和网络环境中情况有些不同。首先是访问者和被访问对象不在一台主机上,它们之间的通信路径可能很长并且中间可能涉及到很多台主机,这些主机的可信赖程度是不同的。因此在进行身份认证时必须将远程用户和本地用户加以区分,在设置访问控制权限时也要对这两种用户区别对待。例如有些资源只允许用户在本地进行访问。其次是规模不同,网络系统的规模比集中式系统要大很多,因此不可能由单台主机来负责管理所有用户以及他们的访问控制信息。必须有某种机制来保证引用监视器与这些用户管理和访问控制信息管理的服务器之间的安全通信。这里涉及到访问控制信息数据完整性和对访问控制服务器的认证协议等问题。

为了简化管理,访问者通常被分成组或组织,设置访问控制时可以按组进行设定,这样就可以避免访问控制表过于庞大。

授权控制框架是对网络资源进行授权管理和访问控制的基本框架。它独立于各应用系统。网络资源包括信息资源和服务资源,如何对这些资源实施统一管理,就需要一个独立于各种应用系统,独立于各个安全子系统的授权管理系统,该系统提供网络资源管理的最主要安全机制。授权控制框架可对各种应用服务进行授权管理,包括万维网 WWW 应用、客户-服务器应用、TCP/IP 应用、数据库应用、面向对象的分布系统应用(CORBA)、报文队列(MQ)应用等标准应用对象。授权管理系统提供的基本服务是管理和维护授权策略、对象映射、用户角色等,并且要有方便使用的管理界面,可以进行安全的远程管理。应用服务系统通过授权应用程序接口获取授权信息,实施用户对对象的访问控制。授权控制框架也应基于国际标准以保证它的互操作性。

参考文献

1. Lampson A. Protection. ACM Operating System Review. 1974, 18(1)
2. Lampson A, Abadi M, Burrows M and Wobber

E. Authentication in Distributed Systems: Theory and Practice. Proceedings of the 13th ACM Symposium on Operating System Principles. Oct. 1991 (胡道元)

fei chuantong jisuanji

非传统计算机(non-traditional computer)

不是采用冯·诺依曼结构的电子计算机。其工作原理不同于传统的冯·诺依曼计算机,也不同于在冯·诺依曼计算机基础上发展起来的其他计算机。

发展简史

冯·诺依曼等人于1945年到1946年间提出了一种计算机的系统结构,这种系统结构采取了存储程序方式,指令和数据一起存在存储器中,存储器按地址访问,在CPU的控制器中设有一个指令计数器,指令按指令计数器指示的顺序逐条地串行执行,后人称这种系统结构为冯·诺依曼结构。按这种结构做成的计算机称为冯·诺依曼计算机。

其后,随着计算机器件、硬件和软件的发展,对最初的冯·诺依曼系统结构作了很多改进,改进的主要目的是要增加计算机的并行处理能力,提高处理速度。采取的主要手段是时间上的重叠和空间上的并行,可以是:操作间的重叠(运算流水线)、指令间的重叠(指令流水线)、向量执行的重叠(向量处理机)、指令级并行(多功能部件、超标量计算机、超长指令字计算机)、多进程-多任务、多线程、用许多个CPU并行工作(多处理机)和分布式处理(网络计算、机群)等。经过这些改进后,指令计数器多了,指令也不再是逐条地串行执行了。但是它们都没有改变主存储器按地址访问,指令执行的次序由指令计数器控制的基本原理,这些计算机都称为控制驱动的计算机。所谓控制驱动,就是在计算机中,只有当指令计数器指向某条指令时才驱动该条指令的执行。控制驱动的计算机称为传统计算机,即冯·诺依曼计算机。

为了适应对计算机运算速度不断提出的更高要求,从20世纪60—70年代起,有人提出多种摆脱冯·诺依曼计算机原有模式束缚的设想,以求在计算机系统结构方面取得更大的突破,开发出具有更高并行功能和系统效率的非传统计算机。为此进行了一系列科研和实践,到80年代已发展出多种类型的非传统计算机。在这些非传统计算机中,有的摆脱了控制驱动结构的束缚,采取了数据驱动结构(数据流计算机采用这种结构)或需求驱动结构(归约机采用这种结构);有的突破了存储器按地址访

问的框框,采取了按内容访问存储器(数据库机主要采用这种结构);有脱离了以数值处理为主的传统计算机模式,开创以符号处理为主的面向智能知识信息处理的智能计算机,等等。

基本内容

在摆脱控制驱动结构的束缚方面 在实际程序中可能存在着大量能够并行执行的指令,但在传统的冯·诺依曼计算机中,由于受到了控制驱动下的串行执行的限制,难以并行执行这些可以并行执行的指令。在数据流计算机(参见**数据流计算机**)中取消了指令计数器,采取数据驱动方式启动指令的执行。数据驱动的含义是:程序中的任一条指令只要其所需的操作数已经全部齐备,就可以立即启动执行。一条指令的运算结果又流向下一条指令作为下一条指令的操作数来驱动该指令的执行。这样,所有有条件执行的指令都能并行执行,而不必等待指令计数器的驱动(其前提是有足够多的硬件执行部件等资源)。如能这样,显然其并行处理能力会远高于传统的计算机。

需求驱动的系统结构与数据驱动的系统结构有些相似,但它们执行操作的次序不同。在数据驱动系统结构中,指令在它的全部输入操作数到齐时立即开始启动执行,这样做的结果难免要多执行很多本来不需要执行的指令。在需求驱动系统结构中,程序仅在需要用它的输出结果时才开始启动。如果此时这条指令的输入数据还未获得,则由这条指令再去启动能获得它所需的各个输入数据的指令,这样就可以把需求链一直延伸下去,直到需要的外部输入数据到达为止。当需求的数值都已到齐,需要它的指令才能继续执行下去,这就是需求驱动的含义。采用需求驱动可以使计算机只需要执行最低限度的计算工作,从而提高计算机的工作效率。归约机就是这样一种需求驱动的计算机(参见**归约机**)。

在以符号处理为主的智能计算机方面 与数值处理比较,符号处理的主要特点有:用符号的知识表示,密集的搜索操作,庞大的存储容量而且不存在访问局部性,消息长度可变,采用不确定算法,用交互的I/O和对知识库的需要。传统计算机的设计适合于作数值计算,不适应符号处理,在进行符号处理时效率低下。因而人们提出了以符号处理为主的智能计算机的设想。到20世纪80年代已研制的智能计算机可分为3类:基于语言的智能机、基于知识的智能机、智能化I/O接口机。

(1) 基于语言的智能机的设计目标是高效地执

行面向人工智能的高级程序设计语言。机器中设有专门的硬件来实现所支持的语言的基本操作,大多是单处理机或只用少数几台高性能处理机组成的粗粒度并行的系统。这类机器又可分为3种:LISP机、Prolog机、函数式语言的智能机。LISP和Prolog是两种最主要的应用于人工智能的高级程序设计语言。

LISP是以表作为处理对象的程序设计语言(参见**LISP语言**)。LISP机是专为支持LISP语言而研制的计算机,它的指令集与LISP语言的基本函数相接近,可以直接执行用LISP语言编写的程序。它对指令集中出现频度高的操作尽量加快执行速度,并采用面向栈的有标志位的数据格式,对LISP语言所具有的动态类型检查、动态嵌套、无用存储空间回收和大量函数调用的特性,在系统结构上都有所考虑,使它在执行LISP程序时达到很高的运行速度。LISP机的研究始于1975年的CONS机,第一台商品化的LAMBDA LISP机出现于20世纪80年代初。80年代在市场上还曾有过多种LISP机产品,较重要的有Symbolics 3600系列、TI Explorer、Xerox 1100系列等。

PROLOG是一种顺序逻辑程序设计语言(参见**PROLOG语言**),PROLOG机是直接执行用PROLOG语言编写的程序的计算机,它以模式搜索和合一为基本机能,以归结原理为基础进行逻辑推理。80年代日本在“第五代计算机系统”计划中曾研制过的**逻辑推理机**即属于PROLOG机。

冯·诺依曼计算机中使用的传统程序设计语言,像FORTRAN、PASCAL等都是命令式语言。由于命令式语言限制了计算机并行性的开发。命令式语言的结构必须服从于串行的冯·诺依曼计算机系统结构,难以全面反映所解题目的本质。函数式语言就是为解决此问题而提出的。函数式程序设计语言需要有适合于它的全新的系统结构。归约机就是一种函数式语言计算机。

(2) 基于知识的智能机的关键部分是知识的表示与操作。为了模拟人脑对知识的并行处理过程,这类机器中包含数量众多的处理器,每个处理器的硬件都比较简单并有一个小容量的局部存储器,通过大量处理器进行高度并行的工作来实现对知识的智能化处理。基于知识的智能机又可分为:语义网络、基于规则的系统、基于目标的系统和神经网络。

(3) 智能化I/O接口机的功能是提供人工智能

系统需要的语言识别、自然语言理解、模式识别、图像处理、计算机视觉等人机间的智能化接口。

在上述这些智能机中,除了有一些 LISP 机和极少数面向人工智能的机器(如 connection machine)有商品化的产品外,其余的都是大学或工业部门的研发实验室中的探索性系统。

在数据库机和知识库机方面 传统计算机不适合于数据库应用中所需的查找、排序、检索、更新、插入、删除以及数据转移等操作,这些操作使数据库管理系统软件的结构复杂,效率不高。20 世纪 70 年代初,随着数据库管理系统(特别是关系型数据库)的推广应用,开始了数据库机的研究。数据库机是一种专门用来替代由数据库管理系统软件所实现的大部分或全部功能的计算机或协处理器。在数据库机中,设置了专门的硬件来提高数据库服务的速度,如外存储器采用关联存储方式以打破磁盘输入输出的瓶颈,用硬件直接进行连接和排序等操作以提高处理速度,采用功能分布式系统进行并行处理等。数据库机曾有过多项实现方案,值得注意的是在网络环境中或具有客户-服务器计算模式的分布式处理环境中的数据库服务器,它是专门用来为客户提供数据库服务的独立的计算机(参见服务器)。

知识库机是一种专门针对知识库管理和知识处理的特点而设计的计算机。知识库机通常总是和智能计算机联系在一起,往往隐含在非冯·诺依曼计算机中。如果把数据库中储存的内容看作是一类断言性知识,就可以把数据库机作为知识库机的组成部分。

展 望

进入 20 世纪 90 年代后,传统计算机的工作频率、存储容量和存取速度、并行处理和网络通信能力都有长足的发展,很多当年想通过突破传统计算机的束缚来达到高性能目标的应用项目已经能在传统计算机的平台上高效率、低成本地完成。高性能的数据库服务器已足以满足共享网络信息资源的需要。因此数据驱动和需求驱动的计算机以及数据库机和知识库机的研制工作基本上都停顿下来。同时, LISP 语言本身有了很大的发展,扩充了新的功能, LISP 机的研究趋向于在传统的通用计算机平台上改进编译技术,增加专用部件或适当修改某些硬件,以达到高性能、低成本运行 LISP 程序的目的,已不再研制专门的 LISP 机。但是非传统计算机的某些研究成果在后来的传统计算机中得到应用。例如,数据驱动原理已用到超标量计算机的乱序发送

中;按内容访问的技术已用到高速缓冲存储器中。

在智能机方面,从 90 年代开始,神经网络得到了很大的发展。在人工神经网络的基础上构建了神经处理机(参见神经计算机)。人工神经网络已能有效地在投资分析、签名分析、过程控制、飞船和机车发动机监控等项目中得到应用。在语言识别、图像识别、工业机器人、医学图像处理、数据采集等领域,都有卓越的表现。

传统的微电子技术集成度和工作频率方面已发展到接近于其物理极限。为适应未来继续发展的需要,人们还在研究生物分子器件(参见生物计算)、光器件(参见光计算机)和量子器件(参见量子计算),用这些全新的器件有可能组成生物计算机、光计算机和量子计算机。

(孙强南)

feidandiao luoji

非单调逻辑 (non-monotonic logic) 引入新公设(或前提)后可能会使原有定理无效的逻辑。像命题逻辑或一阶谓词逻辑这类传统逻辑都具有“单调性”,即当公设或前提增多时,所能推出的结果(即定理)可能随之增多,但决不会减少。非单调逻辑则不然,当公设或前提增多时,所推出的结果(即定理)有可能减少。例如,我们知道“鸟一般是会飞的”,当告诉我们“a 是一只鸟”时,我们便会得出“a 会飞”的结果来。当又告诉我们“b 是一只鸵鸟”,而我们又不知道鸵鸟是不会飞的,便又会得出“b 会飞”的结果。但当告诉我们“鸵鸟不会飞”后,便会撤销“b 会飞”这一结果。

随着认知科学和人工智能的发展,人的常识和人的常识性推理越来越成为突出的问题。常识推理与传统逻辑的演绎推理之根本区别是常识推理具有不确定性,即常识推理并不都是在完全确定的信息状态下进行的。在问题求解过程中,当出现与默认信息相违背的信息时,人们能制定相应的可能方案,但在违背信息出现之前,人们常继续问题的求解。因为传统逻辑难以解决常识推理问题,人们就从非单调逻辑方面找出路,非单调逻辑的研究大约始于 20 世纪的 70 年代末,80 年代初就已出现了多种非单调逻辑系统,迄今为止,比较重要的非单调逻辑系统有基于一阶逻辑的约束化理论、基于模态语言的自动认知理论和介于二者之间的默认推理逻辑。

因为人工智能从一开始就致力于不完全、不确定情形的问题求解,它着重于系统行为的合理性,而不过分追求解的最优或精确,所以非单调逻辑越来越

越受到人工智能研究人员的重视。(王兵山)

fei dandiao tuili

非单调推理 (non-monotonic reasoning)

具有非单调特征的推理,是常识推理和人工智能应用系统中的一种重要的推理方式。

经典逻辑,诸如命题逻辑和一阶逻辑等,具有如下的重要性质:设 P 是推理的前提集合, C 是由 P 导出的结论集合,在向 P 增加了新的前提 p 之后,设由 $P \cup \{p\}$ 导出的结论集合是 C_1 ,则 C 必是 C_1 的子集。换句话说,在向前提集合增加了新的前提后,只可能导出更多的结论,决不能取消或修改原先得到的结论。经典逻辑中推理的这种性质称为推理的单调性。数学中的推理是典型的具有单调性的推理。但是,基于经典逻辑的推理是人们推理的理想化模型,在日常生活中或是在某些人工智能应用系统中,人们经常要依据某些一般来说是正确的但并非绝对正确的规则进行推理,或者在信息不完全的情况下进行推理,这种推理所得的结论是暂时的,可能会修改的,因而不具有单调性,因此人们称之为非单调推理。

对非单调推理作深入的研究是十分必要的。在一阶逻辑中,我们用 $\forall x P(x) = 1$ 表示“所有 x 都具有性质 P ”这一事实。可是实际生活中,这类句子都是近于真实而不是绝对正确的,即大多数 x 具有性质 P ,但偶然也可能会遇见某些例外。例如,所有的鸟儿都能飞,但企鹅和鸵鸟等例外。所有的橘子是黄的,但未熟的和变异的品种例外。由于这类综合性概括语句不是绝对正确的,采用这些语句进行的推理也不可避免地要产生错误。解决这个问题的一种办法是完全抛弃这类语句,这样虽然不会产生错误,但同时也失去了近于真实的东西和许多本来可以得到的结论。另一种办法是修改这类语句,待它完全正确时再使用,可是这种修改相当困难,即使修改好了,句子的结构已变得相当复杂,无法灵活地使用。一种简便而又妥善的处理办法就是先假定这类语句是正确的,并依据它们进行推理,如果在获取了新的事实后发现原来的结论有问题,再取消或修改这些结论,这样一来,推理就具有了非单调性。

为了使非单调推理得到强有力的逻辑支持,人们开始对非单调推理的形式化方法加以认真地研究,提出了各种不同的非单调逻辑。其中较为著名的有 R. Reiter 的默认逻辑, J. McCarthy 的限制逻辑以及 R. C. Moore 的自认知逻辑等等。作为非

单调推理的例子,下面我们着重介绍它的一种重要形式——默认推理。

默认推理与传统推理的根本区别是在推理的前提中增加了如下形式的默认规则:

$$\frac{\alpha(\bar{x}) : M\beta_1(\bar{x}), \dots, M\beta_m(\bar{x})}{r(\bar{x})} \quad (1)$$

其中 $\alpha(\bar{x}), \beta_1(\bar{x}), \dots, \beta_m(\bar{x})$ 和 $r(\bar{x})$ 是一阶逻辑公式,公式中的变元是 $\bar{x} = (x_1, \dots, x_n)$ 。 $\alpha(\bar{x})$ 称为默认规则的前提条件, $\beta_1(\bar{x}), \dots, \beta_m(\bar{x})$ 称为默认规则的理由, $r(\bar{x})$ 称为默认规则的结论。

式(1)的默认规则可以直观地解释为“对于个体 $\bar{x} = (x_1, \dots, x_n)$ 来说,如果 $\alpha(\bar{x})$ 是可相信的, $\beta_1(\bar{x}), \dots, \beta_m(\bar{x})$ 与可相信的公式集合是相容的,则 $r(\bar{x})$ 是可相信的。

默认规则的形式虽然简单,但是却具有很强的表达能力,很多非单调推理过程都可以用默认规则表示出来。例如,在日常生活中,如果有人告诉我说 Tweety 是一只鸟,我们一般总假定它是能飞的,并且以此为结论指导以后的推理与行动,这种推理过程就是使用常识的推理。所依据的常识是“一般说来,鸟是能飞的”。这种常识是不能使用一阶逻辑表示成 $\forall x (Bird(x) \rightarrow Fly(x))$ 的。因为有一些鸟,例如企鹅、鸵鸟等不能飞。使用默认规则可以把这条常识自然地表示成

$$\frac{Bird(x) : M Fly(x)}{Fly(x)}$$

由此可见,默认推理能很好地解决某些常识推理问题。

在知识库系统中,我们经常使用“闭世界假定”约定。即如果由知识库系统得不到正面基本事实 A ,则假定 $\sim A$ 。例如,在航空订票系统中,如果用 $Connect(x, y)$ 表示有连接城市 x 和 y 的直通班机,如果我们在数据库中查不到 $Connect(B, C)$,则回答 $\sim Connect(B, C)$,即城市 B 和 C 之间没有直通班机。这种解决问题的方式称为“把失败当作否定”。闭世界假定的优点在于把知识库系统所需要存储的事实规模减少到最低限度,使用默认规则可以把航空订票系统的闭世界假定描述为:

$$\frac{M \sim Connect(x, y)}{\sim Connect(x, y)}$$

默认推理的前提是由两个集合组成的偶对构成,一个是通常的一阶逻辑公式集合 W ,表示客观事实或绝对正确的规律;另一个是默认规则集合 D ,表

示一般说来是正确的推理规则。默认推理的前提 $T = \langle D, W \rangle$ 也称默认理论。例如前面提到的使用常识推理的例子可以用默认理论 $\Delta_1 = \langle D_1, W_1 \rangle$ 描述, 其中

$$W_1 = \{ \text{Bird}(\text{Tweety}) \}$$

是一阶逻辑公式集合;

$$D_1 = \left\{ \frac{\text{Bird}(x) : \text{MFLy}(x)}{\text{Fly}(x)} \right\}$$

是默认规则集合。这个集合只包含一条默认规则。从默认理论 Δ_1 出发, 可以得到 $\text{Fly}(\text{Tweety})$, 即 Tweety 能飞, 但是这个结论是暂时的, 可以撤回的。如果一个默认理论的所有默认规则中无自由变量, 则该理论称为闭默认理论。

从一个默认理论推出的所有公式集合称为该默认理论的外延, 外延是默认推理中的重要概念。因为在默认规则中涉及对所相信的公式的相容性, 所以在外延的严格的形式化定义中, 使用了算子 Γ 的不动点的概念。设 S 是语言 L 的一个闭公式集合, $T = \langle D, W \rangle$ 是一默认理论, 对 S 使用算子 Γ 的结果是满足下面 3 条性质的最小集合:

性质 1 $W \subseteq \Gamma(S)$

性质 2 $Th(\Gamma(S)) = \Gamma(S)$

性质 3 如果

$$\frac{\alpha : M\beta_1, \dots, M\beta_m}{\gamma} \in D$$

并且 $\alpha \in \Gamma(S)$, $\sim\beta_1, \dots, \sim\beta_m \notin S$, 则 $\gamma \in \Gamma(S)$ 。

如果 E 是算子 Γ 的不动点, 即 $\Gamma(E) = E$, 则 E 称为默认理论 T 的一个外延。

一个默认理论可能有一个或多个外延, 也可能没有外延。例如默认理论 $\Delta_2 = \langle D_2, W_2 \rangle$, 其中

$$W_2 = \{ B \rightarrow \sim A \wedge \sim C \}$$

$$D_2 = \left\{ \frac{MA}{A}, \frac{MB}{B}, \frac{MC}{C} \right\}$$

有两个外延, 它们是 $E_1 = Th(W \cup \{A, C\})$, $E_2 = Th(W \cup \{B\})$ 。

默认理论 $\Delta_3 = \langle D_3, W_3 \rangle$, 其中

$$W_3 = \emptyset$$

$$D_3 = \left\{ \frac{MC}{\sim D'}, \frac{MD}{\sim E'}, \frac{ME}{\sim F} \right\}$$

只有一个外延, $E = Th(\{\sim D, \sim F\})$ 。

默认理论 $\Delta_4 = \langle W_4, D_4 \rangle$, 其中

$$W_4 = \emptyset$$

$$D_4 = \left\{ \frac{MA}{\sim A} \right\}$$

没有外延。

设 $\Delta = \langle D, W \rangle$ 是闭默认理论, E 是 Δ 的一个外延, 则集合

$$\left\{ \frac{\alpha : M\beta_1, \dots, M\beta_m}{w} \in D \mid \alpha \in E, \right. \\ \left. \text{且 } \sim\beta_1, \dots, \sim\beta_m \notin E \right\}$$

称为外延 E 的关于默认理论 Δ 的产生默认规则集, 用 $GD(E, \Delta)$ 表示。

对于默认规则集 D 来说, 集合

$$\left\{ \omega(x) \mid \frac{\alpha(x) : M\beta_1(x), \dots, M\beta_m(x)}{\omega(x)} \in D \right\}$$

称为结论集合, 用 $\text{CONSEQUENTS}(D)$ 表示。

关于产生默认规则集, Reiter 证明了下述重要定理:

设 $\Delta = \langle D, W \rangle$ 是一个闭默认理论, E 是 Δ 的一个外延, 则

$$E = Th(W \cup \text{CONSEQUENTS}(GD(E, \Delta)))$$

这个定理说明了产生默认规则集和外延的关系, 揭示了默认理论外延的结构。

如果一个默认规则具有

$$\frac{\alpha(x) : M\omega(x)}{\omega(x)}$$

的形式, 则将其称为正规的默认规则。现实生活中的绝大多数非单调推理都可以用正规默认规则表示, 如前面所提到的鸟能飞的例子和闭世界假定例子中的默认规则都是正规默认规则。如果一个默认理论中的所有默认规则都是正规默认规则, 则该默认理论称为正规默认理论。正规默认理论有许多好的性质, 列举如下:

性质 1 每一闭正规默认理论都有外延。

性质 2(半单调性) 设 D 和 D' 是两个闭正规默认规则集, 并且 $D' \subseteq D$, $\Delta' = \langle D', W \rangle$ 和 $\Delta = \langle D, W \rangle$ 是两个闭正规默认理论, 又设 E' 是 Δ' 的一个外延, 则 Δ 存在外延 E , 满足:

$$\textcircled{1} E' \subseteq E;$$

$$\textcircled{2} GD(E', \Delta') \subseteq GD(E, \Delta);$$

即 E' 的关于 Δ' 的产生默认规则集是 E 的关于 Δ 的产生默认规则集的子集。

性质 3(外延的正交性) 如果一个闭正规默认理论 $\langle D, W \rangle$ 有两个不同的外延 E 和 F , 则 $E \cup F$ 是不相容的。

性质 4 设 $\Delta = \langle D, W \rangle$ 是一个闭正规默认理论, 并且 $W \cup \text{CONSEQUENTS}(D)$ 是相容的, 则 Δ 有

惟一的外延。

性质 5 设 D' 和 D 是两个闭正规默认规则, 且 $D' \subseteq D, \Delta = \langle D, W \rangle$ 和 $\Delta' = \langle D', W' \rangle$ 是两个闭正规默认理论; 又设 E'_1 和 E'_2 是 Δ' 的两个不同的外延, 则 Δ 有两个不同的外延 E_1 和 E_2 满足 $E'_1 \subseteq E_1$ 且 $E'_2 \subseteq E_2$ 。

Reiter 不但对默认推理给出了有效的形式化描述, 而且对闭正规默认理论给出了证明理论。证明理论的主要目的是回答, 是否可以相信一个一阶逻辑公式的问题。换句话说, 即是要完成下述判定任务: 给定一个闭正规默认理论 Δ 和一个闭一阶公式 β , 问 Δ 是否具有包含 β 的外延。

设 D 是一个有限闭正规默认规则集, 则集合

$$\left\{ \alpha \mid \frac{\alpha : M\omega}{\omega} \in D \right\}$$

称为 D 的前提条件集合, 记为 $\text{PREREQUISITES}(D)$ 。

设 $\Delta = \langle D, W \rangle$ 是闭正规默认理论, β 是闭一阶公式, Reiter 把 β 的关于 Δ 的默认证明定义成有限默认规则子集序列 D_0, \dots, D_k , 满足如下 4 个条件:

- ① $W \cup \text{CONSEQUENTS}(D_0) \vdash \beta$;
- ② 对于所有的 $i (1 \leq i \leq k)$
 $W \cup \text{CONSEQUENTS}(D_i) \vdash$
 $\text{PREREQUISITES}(D_{i-1})$;
- ③ $D_k = \emptyset$;
- ④ $W \cup \bigcup_{i=0}^k \text{CONSEQUENTS}(D_i)$ 是可满足的。

Reiter 还证明了关于默认证明的可靠性与完备性结果。可靠性结果告诉我们, 如果 $\Delta = \langle D, W \rangle$ 是闭正规默认理论, β 是闭一阶公式, 且 β 有一个关于 Δ 的默认证明 D_0, \dots, D_k , 则 Δ 有包含 β 的外延 E 。完备性结果告诉我们, 如果 $\Delta = \langle D, W \rangle$ 是闭正规默认理论, β 是闭一阶公式, 并且 Δ 有包含 β 的外延 E , 则 β 有关于 Δ 的默认证明。把关于默认证明的可靠性与完备性结合起来, 则知道闭正规默认理论 Δ 是否有包含 β 的外延问题就是 β 是否有关于 Δ 的默认证明的问题。这个结果是令人满意的, 但是要获得一个闭公式关于某个正规默认理论的证明仍是一个十分困难的问题。Reiter 证明了关于某一闭公式 β 是否有关于闭正规默认理论 Δ 的证明问题 (默认证明存在问题) 甚至不是半可判定的, 也就是说, 即便 β 有关于 Δ 的默认证明, 也不能保证肯定能得到这个证明。我们知道, 从任何一阶逻辑公式集合能够推导出的定理集合是递归可数的, 如果把默认

理论的外延包含的所有一阶公式看作是默认理论的“定理”, 则 Reiter 的关于默认证明存在问题不是半可判定的结果告诉我们, 任意默认理论的“定理”不是递归可数的, 这也是默认逻辑与传统逻辑的一个重要区别。

虽然闭正规默认理论的默认证明存在问题不是半可判定的, 但是对于闭正规默认理论的某些具体问题来说默认证明存在问题却是可判定的。为解决这类判定问题, Reiter 提出了自顶向下的判定算法。这个算法的主要思想是: 对于给定的闭一阶公式 β , 确定默认规则集的一个子集 D_0 , 使 $W \cup \text{CONSEQUENTS}(D_0) \vdash \beta$, 对于 $i \geq 1$, 如果已经确定了 D_{i-1} , 则确定默认规则的子集 D_i , 使得

$$W \cup \text{CONSEQUENTS}(D_i) \vdash \text{PREREQUISITES}(D_{i-1})$$

如此反复继续下去, 如果存在某一 k , 使得

$$W \vdash \text{PREREQUISITES}(D_{k-1})$$

$k-1$

并且 $W \cup \bigcup_{i=0}^{k-1} \text{CONSEQUENTS}(D_i)$ 是可满足的, 则我们就发现了 β 的一个证明。

为了实现上述想法, 首要的问题是如何选择 D_0 。如果随机地在默认规则集 D 中选择子集 D_0 , 然后检测从 $W \cup \text{CONSEQUENTS}(D_0)$ 是否能推导出 β , 这种方法的计算量显然是相当大的, 因而是不可取的。Reiter 利用线性归结的思想, 引进一种特殊的子句形式——标记子句, 巧妙地使用 β 选出了 D_0 。

设 $\langle D, W \rangle$ 是一闭正规默认理论, C_1, \dots, C_r 是所有的由默认规则的结论产生的子句, 则 C_i 与产生 C_i 的默认规则 δ 组成的偶对 (C_i, δ) 称为结论子句, 把由 D 产生的所有结论子句和由 W 产生的所有子句的偶对形式 $\{(C_i, \delta) \mid C_i \in W\}$ 放在一起组成了标记子句, 设 (C_1, D_1) 和 (C_2, D_2) 是两个标记子句, R 是 C_1 和 C_2 的通常的归结式, 则 $(R, D_1 \cup D_2)$ 称为 (C_1, D_1) 和 (C_2, D_2) 的归结式。仿照对公式 β 的线性归结证明的概念, Reiter 给出了关于 β 的从标记子句出发的线性归结证明的概念, 并且证明了如果存在这样的线性归结证明, 则

$$W \cup \text{CONSEQUENTS}(D_0) \vdash \beta$$

从而解决了 D_0 的选取问题。选出 D_0 之后, 可以把 $\text{PREREQUISITES}(D_0)$ 当作顶部的初始子句, 再次使用基于标记子句的线性归结, 求出 D_1 , 如此反复进行下去, 直到获得 β 的默认证明。

非单调推理与人们的通常推理过程十分接近, 具有较高的实用价值, 引起很多人的兴趣与重视。国际人工智能联合会 (IJCAI)、美国人工智能协会

(AAAI)以及国际人工智能杂志都曾为非单调推理开辟过专题,形成了研究非单调推理的热潮。非单调推理在逻辑程序设计、智能数据库与诊断推理等领域有广泛的应用。

参考文献

Reiter R. Nonmonotonic Reasoning. In: Annual Reviews in Computer Science 2; Annual Reviews Inc., 1987. 147 ~ 186
(姜云飞)

feiguocheng yuyan

非过程语言 (nonprocedural language) 不显式指明处理过程细节的**程序设计语言**。这里所说的“处理过程细节”,不是指待解问题及其解法的本质所要求的(问题逻辑所固有的),而是指为计算机上实现求解任务而设定的计算细节及其执行顺序。

在传统的**高级语言**中,程序一般由描述处理对象的一组说明和描述处理动作及执行次序的一组**语句**组成。如果一种语言的基本成分的抽象级较高,那么它的描述能力就较强,在抽象级相对较低的语言中需要用许多语句才能表示的计算过程细节,在抽象级高的语言中往往可以用一条语句加以概括,以更简明直接的形式表示出来。例如,APL语言中用一条语句可以直接表示两个矩阵相乘的计算任务(如 $A \leftarrow B + . \times C$),但在**FORTRAN语言**中对同一计算要求,要用三重循环写成的一段程序来表示具体的计算过程。这就是说,语言的抽象级越高,则用它编写的程序中有关计算过程细节的描述就越少,相应地,语言的“非过程性”体现得也更为明显。因此,在一定意义上说,非过程语言的含义是相对的。在许多情况下与其说一种语言是非过程的,不如说语言的过程性特征较少。

目前一般认为非过程语言的主要特征有如下三点:①数据的联想引用机制;②数据的高级操作符;③不指明可变性顺序。联想引用机制是指,不指明访问路径、索引或其他结构位置信息,而只需给出该数据应满足的条件或性质的形式描述就可以存取数据的机制;高级操作符是指可直接表示集合、数组、表等复杂数据运算的操作符;可变性顺序是指改变执行次序也不影响处理结果的顺序,它不是问题求解方法的本质所确定的,而是编写程序时因语言描述能力的限制而程序设计人员加以精化和规定的顺序。

早在20世纪60年代,就已开始以冯·诺依曼式计算模型为基础的非过程语言的研究。针对数据

处理领域中的特定问题,曾出现了一些具有非过程特征的语言,例如:描述判定表的语言DETAB-65,报表程序的生成器RPG等。其中DETAB-65不显式指明判定表中规则的执行顺序(可变性顺序),RPG只需输入关于输出表格信息即可。但这些语言中有关数据运算描述的抽象级较低。而对通用型非过程语言的研究工作,由于它不仅与语言处理技术直接相关,而且与自动程序设计技术也有密切联系,所以进展比较缓慢。迄今为止,设计这类语言时采用的主要办法是,在语言中尽量引进各种抽象度较高的非过程性描述手段,以期做到在程序中增加“做什么”的描述成分,减少“如何做”的细节描述,SETL和第四代语言(4GL)有其一定代表性。SETL是以集合论为基础的超高级语言。为了提高程序的抽象级别,它引进集合作为数据类型,并提供“联想引用”机制和可以直接表示集合运算的很多高级运算符。第四代语言是20世纪80年代出现的新概念。虽然目前还没有公认的严格定义,但一般认为它是抽象程度很高、非过程性很强的一类语言的统称,例如,各种数据库查询语言、程序生成器和能转换成可执行代码的形式规约语言等,它们一般都具有联想引用机制和高级操作符。

在70年代以后出现的**逻辑式程序设计语言**和**函数式程序设计语言**是以非冯·诺依曼式计算模型为基础的新型语言。逻辑式程序设计语言的典型代表是**PROLOG语言**。语言的基本成分是HORN子句,程序则由用它描述的问题求解所需的一系列事实与推理规则组成。程序的执行过程是从给定目标出发,利用程序中的事实和规则,反复进行合一、归结和回溯的过程。然而,这个执行过程基本上没有显式地反映在程序之中。这就是说,PROLOG语言不仅具有联想引用机制,而且还有不指明可变顺序特征。函数式程序设计语言以函数作为程序的基本成分,并提供一系列用来构造更为复杂函数的定义设施,程序就是根据给定问题使用这些设施构造的求解函数的表达式。由于语言中没有赋值语句,没有副作用,因此如果函数有定义,则计算结果与表达式的计算次序无关。
(金淳兆)

fei jidashi yinshuaji

非击打式印刷机 (nonimpact printer) 一种利用物理的(光、电、热、磁)或化学的方法,印字头不与纸或其他媒体接触,或虽有接触但无击打动作的印刷设备。非击打式印刷机有许多类型,它们是

基于热敏、喷墨、电灼、静电、电子照相转印以及其他如离子沉积、磁化成像转印等原理制成的。由于这类设备不需击打动作,因而它的工作噪声低,印字速率快。又由于计算机所用的各种非击打式印刷设备都属点阵型,因而字体变换方便,字符种类不受限制,适于汉字印刷,可印图形及图像。

喷墨印刷机 多采用随机喷墨方式,在需要时才从喷嘴喷射墨滴。印字头由一系列喷嘴构成,每个喷嘴均可根据需要分别喷出墨滴。产生墨滴的原理有三种。①用压电传感元件的变形压迫压力腔的内壁,从喷嘴挤出墨滴。②将墨汁加热产生气泡,使墨汁经喷嘴射出。③使用固体色棒,色棒在高压脉冲的作用下溶出墨滴,经喷嘴射出。

利用喷墨技术可实现彩色印刷。在彩色印刷的喷墨印字头上装有青、品红、黄三色及黑色共四套喷嘴和墨盒(照片打印机可多达7色),喷嘴个数多的达到3072个(512×6)。打印分辨率从原来的180 dpi(点每英寸)提高到后来的4800 dpi,墨滴体积由以往的30 pL($1 \text{ pL} = 10^{-12} \text{ L}$)减小到后来的2 pL,4 pL。因此,能印出质量非常高的彩色图文。打印速度达到17 ppm(页每分)(单色,A4)、12 ppm(彩色,A4)。

彩色喷墨印刷机是2003年流行的照片印刷工具。在提高印刷质量的同时,喷墨照片印刷机印刷速度达到1分钟之内印一张4英寸×6英寸的照片。

大幅面(A1以上)彩色喷墨印刷机俗称喷绘机,它在CAD/GIS、广告制作方面有广泛应用。多采用6色墨水,2003年印刷分辨率可达1440 dpi,印刷速度可达720 m^2/h 。

电子照相印刷机 利用电子照相转印的印刷设备。在光电绝缘材料制成的转印鼓(或带)的表面预先施加静电荷,然后,由字符或图形信息调制的光束在其上扫描、曝光。被光照射部分的电荷消失,成像部分的电荷保留下来,形成静电潜像。经过吸附色粉的显影处理后,带电部分吸附上色粉而形成色粉图像。然后,再转印到纸上,热压定影。利用电子照相转印技术制成的印刷设备,因曝光用的光源不同,有激光印刷机、发光二极管印刷机、静电印刷机、液晶快门印刷机、等离子发光管印刷机以及荧光管印刷机等多种类型。其中,激光印刷机的发展历史最久,应用最广。

激光印刷机 利用激光器件作光源的一种电子照相转印型印刷设备。20世纪70年代中期,西门

子公司和IBM公司先后都推出了这种印刷机,每分钟可印字符万行以上,属高速印刷设备。

其工作原理如下:由激光器发出的激光束经声光调制偏转器按字符点阵的信息调制。在高频超声信号的作用下,声光偏转器衍射出形成字符的调制光束。当频率变化时,激光束的衍射角度随之变化,形成纵向的扇出。此扇出光束经高速恒速旋转的多面镜反射,在预先荷电的、用光电特性材料制成的转印鼓面上沿鼓的轴向扫描曝光。鼓面被激光束照射的部位电荷消失,形成静电潜像。当鼓面经过带相反电荷的色粉时,由于静电作用吸附上色粉,进行显影。在电晕电场的作用下,色粉由鼓面被转印到纸上。经热滚挤压定影之后,字符便永久性地印在纸上。

激光印刷速度高,印字质量好,分辨率高,噪声小,可印刷多种类型的字符字体、图形、图像,字形大小变化灵活,适于大量数据的连续印刷。通常激光印刷机以页为单位进行操作,故常称之为页式印刷机。2003年达到的分辨率为2400 dpi,打印速度达37(页每分)(黑白,A3)、30 ppm(彩色,A3)。

发光二极管印刷机 利用发光二极管作光源的电子照相转印型印刷设备。发光二极管阵列构成一横排发光点直接在转印鼓或带的表面上曝光,省却了像激光印刷机所用的复杂的光束偏转扫描系统。发光二极管印字头由多片(一般为30至40片)发光二极管阵列芯片(每片64或128个发光点)、驱动集成电路及对应每个发光点的棒状聚焦透镜组成。

由于这种印刷机使用了高集成电路技术,又无可动部件,因而可靠性高,且结构较激光印刷机简单、紧凑。一行1000~4000发光点同时沿纸的走向扫描,印字头不需作横向往复运动。分辨率可达400 dpi,印刷速度为15~20 ppm(页每分)。

静电印刷机 利用多个电极直接在具有良好绝缘性能的电介质层的特殊纸上施加电荷,形成静电潜像,经过处理而成像的印刷方法。写入电极可以是一横排针或一组扫描针,也可以是针屏阴极射线扫描管上的针屏。当500~1500V的脉冲电压加到针上时,或当阴极射线管内电子束扫描针屏时,针端便在纸的电介质层上产生电荷,形成静电潜像。然后经过干式或湿式显影及热定影等处理,形成永久性的字符。静电印刷机是一种较早出现的非击打式印刷设备。它的结构简单,噪声小,印刷速度快,每分钟可印2万行。但它需用特殊的记录纸,显影与定影过程复杂。

热敏印刷机 用于计算机的热敏印刷技术有热

敏纸式与热转印式两种。

热敏纸印刷机 利用印字头上多个电热元件在特殊的热敏纸上瞬时加热,引起与热元件接触部位纸的变色而形成字符。热印头上一列或多列热元件的发热点与纸保持适当接触,印头自左至右移动,各热元件在字符点矩阵的控制下快速地热冷交替变换。这种印字机的主要优点是机构简单,体积小,噪声小。其主要缺点是热印头的热反应速率不高,热敏纸的字迹保存性差。

热转印印刷机 利用转印色带将字符转印到纸上的印刷设备。在几微米厚的聚酯薄膜上涂以低熔点的固态墨作为转印色带,一排点状电阻发热元件装在陶瓷基片上构成热印头。印刷时,热印头压在色带与纸上并根据字符点阵适时通过脉冲电流,使色料熔化而转印到纸上。转印色带有许多种,常见的为蜡色带、升华带,其他还有热阻带、染料扩散带等。

热升华印刷机 通过半导体加热元件可调节每一点的温度,组合出色彩的比例和浓淡程度,达到连续色调的彩色照片的效果。转印的画面质量可与银盐相片媲美。但打印速度慢,耗材贵,使用环境要求高。2003年有的产品达到334 dpi的彩色分辨率,印刷一张 3.5×5 的照片的时间是26秒。

参考文献

Myers R A, Tamulis J C. . Introduction to Topical Issue on Non Impact Printing Technologies. IBM J. Res. Develop. , 1984, 28(3):234~240 (刘锡刚)

feijiandu xuexi

非监督学习 (unsupervised learning) 在没有类别信息情况下,通过对所研究对象的大量样本的数据分析实现对样本分类的一种数据处理方法。

由于在很多实际应用中,缺少所研究对象类别形成过程的知识,或者为了判断各个样本(模式)所属的类别需要很大的工作量(例如卫星遥感照片上各像元所对应的地面情况),因此往往只能用无类别标签的样本集进行学习。通过无监督学习,把样本集划分为若干个子集(类别),从而直接解决了样本的分类问题,或者把它作为训练样本集,再用监督学习方法进行分类器设计。无监督学习主要有以下两大类方法:

(1) 基于概率密度函数估计的直接方法

如果给定的样本集是由各类都服从高斯分布的样本混合在一起组成的,在类别数已知的条件下,可

以用最大似然法或 Bayes 估计方法,从混合的概率密度函数中分解出各个类的概率密度函数,然后用 Bayes 决策方法设计模式分类器。在非高斯概率分布情况下,只要各类的概率密度函数的形式已知,且分解是惟一的,都可以用上述方法实现分类器设计。在没有任何概率分布先验知识的情况下,可以把特征空间划分为若干个区域,使每个区域都具有单峰的性质,每一个区域就相当于一个类别。这样作的基础是紧致性假设(参见**模式分类器**)。已经有多种算法实现这种区域的划分。

(2) 基于样本间相似性度量的间接聚类方法

如果用样本在特征空间中相互间的距离来度量样本间的相似度,就可以设计出某种评价分类质量的准则函数,通过数学方法把特征空间划分为与各个类别相对应的区域,也就是通常所说的**聚类分析**。有两大类基本的聚类分析算法,即迭代的动态聚类算法和非迭代的分级聚类算法。前者是给定某个样本集的初始划分,计算反映聚类质量的准则函数值。如果把某个样本从原来所属的类别改属为另一个类别能使准则函数值向好的方向改进,则改变这个样本原来的类别为新的类别(新的划分)再对其他样本进行类似的运算。这样反复迭代,直到没有一个样本类别的改变能改进准则函数值,即已经达到了准则函数的最优值。这一类算法中著名的有C-均值算法和ISODATA算法,C-均值算法要求类别数预先给定,并把各样本到所属类别样本子集的均值向量的距离平方和作为评价聚类质量的准则函数。ISODATA算法可以自动地调整类别数,并可对各类样本的某些统计性质(如样本数量、样本特征的标准偏差等)作些限制。非迭代的分级聚类算法:第一步把每一个样本都看成一个类,给定两类样本间相似度计算方法,计算类与类之间的相似度。第二步把其中相似度最大的两个类合并为一个类,再计算新的类与类之间的相似度。第三步再把其中相似度最大的两个类合并为一个类,依此进行下去,直到把所有的样本都合为一类为止。根据问题的性质以及各级的相似度大小,就可以确定合理的聚类差数和各类所包含的样本。在应用分级聚类算法时要选择适当的类与类间相似度计算方法,不同的计算方法会导致完全不同的聚类结果。

聚类分析是无监督学习的主要方法,它能从大量的数据集中找出有规律性的结果。为了适应各种实际问题的数据结构的特点,还发展了以上述方法为基础的各种其他算法。

参考文献

1. Duda R O, Hart P E. Pattern Classification and Scene Analysis. New York: John Wiley & Sons, 1973

2. Devijver P A, Kittler J. Pattern Recognition: A Statistical Approach. New Jersey: Prentice Hall, 1982
(边肇祺)

feixianxing daishu fangchengzu shuzhi
jiefa

非线性代数方程组数值解法 (numerical solution for system of nonlinear algebraic equations) 根据非线性代数方程组特点, 研究如何采用有效的数值方法 (直接法和迭代法) 来求非线性代数方程组

$$f_i(x_1, x_2, \dots, x_n) = 0 \quad i = 1, 2, \dots, n \quad (1)$$

近似解的方法和过程。式(1)中 $f_i(x_1, \dots, x_n)$ 是定义在 n 维空间 R^n 的开域 D 上的实函数, 且至少有一个是关于 x_1, \dots, x_n 的非线性函数。在 R^n 中记 $\mathbf{x} = (x_1, \dots, x_n)^T$, $\mathbf{f} = (f_1, \dots, f_n)^T$, 则方程组(1)简写为 $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ 。若存在 $\mathbf{x}^* \in D$, 使 $\mathbf{f}(\mathbf{x}^*) = \mathbf{0}$, 则称 \mathbf{x}^* 是非线性方程组的解。方程组可能有一个解或多个解, 也可能有无穷多解或无解。除极特殊方程外, 一般不能用直接法求得其精确解, 目前主要采用迭代法求满足精度要求的近似解。迭代法就是根据不同思想构造逐次逼近方程组解 \mathbf{x}^* 的迭代序列

$$\mathbf{x}^{k+1} = \mathbf{g}(\mathbf{x}^k) \quad k = 0, 1, \dots \quad (2)$$

$$\text{或 } \mathbf{x}^{k+1} = \mathbf{g}(\mathbf{x}^k, \mathbf{x}^{k-1}, \dots, \mathbf{x}^{k-m+1}) \quad k = 0, 1, \dots$$

(3)

前者称为单步迭代法, 后者称为多步 (m 步) 迭代法。计算时只要给定初始近似 \mathbf{x}^0 , 则可由式(2)逐次算出 $\mathbf{x}^1, \mathbf{x}^2, \dots$, 当 $\lim_{k \rightarrow \infty} \mathbf{x}^k = \mathbf{x}^*$, 则称迭代序列 $\{\mathbf{x}^k\}_{k=1}^{\infty}$ 收敛于解 \mathbf{x}^* 。如果存在常数 $p \geq 1$ 以及 $c_p > 0$, 使

$$\lim_{k \rightarrow \infty} \frac{\|\mathbf{x}^{k+1} - \mathbf{x}^*\|}{\|\mathbf{x}^k - \mathbf{x}^*\|^p} = c_p \quad c_1 < 1$$

则称迭代序列 $\{\mathbf{x}^k\}_{k=1}^{\infty}$ 对于 \mathbf{x}^* 是 p 阶收敛的, c_p 称为收敛因子, $p = 1$ 为线性收敛, $p > 1$ 称为超线性收敛, 收敛阶 p 越大, 收敛越快, 在 p 相同的情况下, 则 c_p 越小收敛越快。为了评价不同迭代法的优劣, 通常用效率 $e = \frac{\ln p}{w}$ 作为衡量标准, 其中 p 为迭代法的收敛阶, w 为每迭代步计算分量函数值 f_i 及偏导

数 $\frac{\partial f_i}{\partial x_j}$ 的总个数。效率 e 越大, 表示迭代法花费代价越小。

求解非线性方程组最重要的迭代法有牛顿法及其变形、拟牛顿法等。

牛顿法及其变形 牛顿法基本思想是将非线性函数 $\mathbf{f}(\mathbf{x})$ 逐步线性化形成以下的迭代序列:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \left[\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}} \right]^{-1} \mathbf{f}(\mathbf{x}^k) \quad k = 0, 1, \dots \quad (4)$$

其中

$$\left[\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}} \right] = \begin{bmatrix} \frac{\partial f_1(\mathbf{x}^k)}{\partial x_1} & \frac{\partial f_1(\mathbf{x}^k)}{\partial x_2} & \dots & \frac{\partial f_1(\mathbf{x}^k)}{\partial x_n} \\ \frac{\partial f_2(\mathbf{x}^k)}{\partial x_1} & \frac{\partial f_2(\mathbf{x}^k)}{\partial x_2} & \dots & \frac{\partial f_2(\mathbf{x}^k)}{\partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_n(\mathbf{x}^k)}{\partial x_1} & \frac{\partial f_n(\mathbf{x}^k)}{\partial x_2} & \dots & \frac{\partial f_n(\mathbf{x}^k)}{\partial x_n} \end{bmatrix}$$

是 $\mathbf{f}(\mathbf{x}^k)$ 的雅可比矩阵。当 \mathbf{x}^0 是解 \mathbf{x}^* 的一个较好近似时, 牛顿迭代序列(4)是2阶收敛的。由 \mathbf{x}^k 计算 \mathbf{x}^{k+1} 的步骤为: ①计算 $\mathbf{f}(\mathbf{x}^k)$ 及 $\left[\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}} \right]$ 。②用

直接法解线性方程组 $\left[\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}} \right] \Delta \mathbf{x}^k = -\mathbf{f}(\mathbf{x}^k)$, 称为

牛顿方程。③计算 $\mathbf{x}^{k+1} = \mathbf{x}^k + \Delta \mathbf{x}^k$ 。编程上机计算到 $\|\mathbf{x}^k - \mathbf{x}^{k+1}\| \leq \varepsilon$, 或 $\|\mathbf{f}(\mathbf{x}^k)\| \leq \varepsilon$ 停止, 其中 ε 为给定精度。牛顿法的优点是收敛快且可以自修正, 缺点是每步要计算雅可比阵 $\left[\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}} \right]$, 工作量为

$w = n^2 + n$ 。另外, 要求 \mathbf{x}^0 在解 \mathbf{x}^* 附近较难达到。

为放宽牛顿法对初始近似 \mathbf{x}^0 的限制, 扩大收敛范围, 可引进松弛参数 $\omega_k > 0$, 将牛顿法程序改为:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \omega_k \left[\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}} \right]^{-1} \mathbf{f}(\mathbf{x}^k) \quad k = 0, 1, \dots \quad (5)$$

称为牛顿下山法, 参数 ω_k 由条件 $\|\mathbf{f}(\mathbf{x}^{k+1})\| < \|\mathbf{f}(\mathbf{x}^k)\|$ 确定。

为减少牛顿法计算工作量, 提高效率, 可用修正牛顿法, 其迭代公式为:

$$\left. \begin{aligned} \mathbf{x}^{k,i} &= \mathbf{x}^{k,i-1} - \left[\frac{\partial \mathbf{f}(\mathbf{x}^k)}{\partial \mathbf{x}} \right]^{-1} \mathbf{f}(\mathbf{x}^{k,i-1}) \quad i = 1, \dots, m \\ \mathbf{x}^{k,0} &= \mathbf{x}^k, \quad \mathbf{x}^{k+1} = \mathbf{x}^{k,m} \quad k = 0, 1, \dots \end{aligned} \right\} \quad (6)$$

修正牛顿法的收敛阶为 $p = m + 1$, 每迭代步的工作

量为 $W = n^2 + mn$, 方法的效率为 $e_m = \frac{\ln(m+1)}{n^2 + mn}$, e_1 即为牛顿法效率。当 $n = 10, m = 7$ 时, $e_m/e_1 = 1.94$, n 越大 e_m/e_1 越大, 它表明修正牛顿法 (6) 比牛顿法效率高。

计算机上往往采用差商近似偏导数的 **离散牛顿法**:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - [\mathbf{A}(\mathbf{x}^k, \mathbf{h}^k)]^{-1} \mathbf{f}(\mathbf{x}^k) \quad k = 0, 1, \dots \quad (7)$$

式中矩阵 $\mathbf{A}(\mathbf{x}^k, \mathbf{h}^k)$ 的元素 $[\mathbf{A}(\mathbf{x}^k, \mathbf{h}^k)]_{ij} = \frac{f_i(\mathbf{x}^k + h_j^k \mathbf{e}_j) - f_i(\mathbf{x}^k)}{h_j^k}$ ($i, j = 1, 2, \dots, n$), 其中 \mathbf{e}_j 为坐标向量, $\mathbf{h}^k = (h_1^k, \dots, h_n^k)^T$, 这个方法具有超线性敛速, 当 $\mathbf{h}^k = \mathbf{f}(\mathbf{x}^k) = (f_1(\mathbf{x}^k), \dots, f_n(\mathbf{x}^k))^T$ 时, 公式 (7) 称为牛顿-斯蒂芬森方法, 它具有 2 阶敛速。

在牛顿法 (4) 中, 若解牛顿方程组不用直接法, 而采用解线性方程组的迭代法, 则得一类非线性与线性的双重迭代法, 这类方法常用牛顿-SOR 迭代法。此外, 还可将解线性方程组迭代法思想用于解非线性方程组, 得到一类非线性松弛法, 如 SOR-牛顿法, 这类方法优点是程序简单, 存储量省, 但收敛较慢。

拟牛顿法 是一类不用计算 $\mathbf{f}(\mathbf{x})$ 的雅可比矩阵, 又具有超线性收敛的算法。它是 20 世纪 60 年代中期出现的新算法, 有很多不同的计算公式, 其中常用的秩 1 拟牛顿法是 **布罗依登法**, 其计算公式为:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{A}_k^{-1} \mathbf{f}(\mathbf{x}^k)$$

$$\mathbf{A}_{k+1} = \mathbf{A}_k + \frac{[\mathbf{y}_k - \mathbf{A}_k \mathbf{S}_k] \mathbf{S}_k^T}{\mathbf{S}_k^T \mathbf{S}_k} \quad k = 0, 1, \dots$$

其中 $\mathbf{S}_k = \mathbf{x}^{k+1} - \mathbf{x}^k$, $\mathbf{y}_k = \mathbf{f}(\mathbf{x}^{k+1}) - \mathbf{f}(\mathbf{x}^k)$ 。

此外, 还有 **割线法**、**布朗方法**、**布兰特方法**、**连续法**、**区间方法**和**多分裂算法**等多种方法, 其中布兰特方法由于效率较高, 已在数学包中被广泛使用。连续法由于对初始近似 \mathbf{x}^0 没有特别限制而受到重视。区间方法在判断非线性方程解的存在惟一性方面较简便。多分裂算法适用于在并行处理机上计算, 近年有很大发展。

在方程 (1) 中当 $n = 1$ 时, 就是非线性方程 $f(x) = 0$, 前面介绍的牛顿法、牛顿下山法、离散牛顿法、割线法等都是常用的数值方法, 其中割线法又称弦位法, 其迭代公式为

$$x_{k+1} = x_k - \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} f(x_k) \quad k = 1, 2, \dots \quad (8)$$

这是一种多步迭代法, 计算时先给出初始近似 x_0, x_1 , 再按公式 (8) 逐次求出 x_2, x_3, \dots , 每步只算一个新函数值, 计算量较省, 而方法收敛阶为 $p = (1 + \sqrt{5})/2 = 1.618\dots$, 故效率也较高。割线法实际上是一种线性插值法, 如果利用 x_k, x_{k-1}, x_{k-2} 三点造 $f(x)$ 的二次插值, 可得到求非线性方程的抛物线法, 其迭代公式为

$$x_{k+1} = x_k - \frac{2f(x_k)}{b_k \pm \sqrt{b_k^2 - 4f(x_k)f[x_k, x_{k-1}, x_{k-2}]}} \quad k = 2, 3, \dots$$

式中 $b_k = f[x_k, x_{k-1}] + f[x_k, x_{k-1}, x_{k-2}](x_k - x_{k-1})$, “ \pm ”号选取与 b_k 同号, $f[\cdot, \cdot], f[\cdot, \cdot, \cdot]$ 分别表示 $f(x)$ 在相应点的一阶与二阶差商, 抛物线法每步也只算一个新函数值 $f(x_k)$, 其收敛阶为 $p = 1.839\dots$, 效率比割线法又有提高, 且可求方程的复根, 因此也是非线性方程数值解的常用算法。

科学和工程计算中经常用到非线性方程和方程组数值解法, 如在各种非线性力学问题、电路问题、经济平衡问题、非线性规划以及非线性微分方程数值解法中都要用到。

参考文献

1. 李庆扬, 莫效中, 祁力群. 非线性方程组数值解法. 北京: 科学出版社, 1992
2. Traub J F. Iterative Methods for the Solution of Equations. Englewood cliffs, New Jersey: Prentice-Hall, 1964

(李庆扬)

fei zhenshigan tuxing huizhi

非真实感图形绘制 (non-photorealistic rendering, NPR) 相对**真实感图形**生成而提出来的绘制方法。其他如 expressive rendering, artistic rendering, painterly rendering, interpretative rendering 等是 NPR 的同义词, 最近 stylised rendering 即风格化绘制逐渐被人们所接受。

早期的非真实感图形绘制 (NPR) 技术主要是模拟传统绘画效果, 如钢笔画、铅笔画、油画、水彩画等, 近年来出现一些 NPR 技术绘制出的图形既有别于真实感也有别于传统绘画效果。在 NPR 技术分类上有基于模型的、基于图像的以及两者混合的, 也可以从 NPR 技术应用到二维或是三维空间以及是否需要交互进行分类。

NPR 技术主要有两部分内容:

- (1) 笔刷生成 它包括笔刷纹理和笔刷形状,

对于传统笔刷纹理可以用模型模拟,也可以用图像合成。在模拟传统笔刷纹理时往往需要考虑构成画笔材料的特性、绘画颜料的特性以及画纸或画布的特性。笔刷形状可以用施加在画笔上力的大小作为输入参数进行计算,也可以固定函数定义。在一些非传统笔刷模型中,其纹理和形状可以有多种变化,如圆弧、螺旋线以及字母等都可以作为笔刷纹理单元。

(2) 笔刷绘制 它包括控制笔刷的方向、长短、粗细以及在二维平面或三维曲面上的分布。在不同技术种类及不同应用中对它们有不同的控制方法。

在基于模型的二维 NPR 技术中需要在二维平面上设计画笔的运动轨迹,轨迹的方向即为笔刷的方向,笔刷的长短、粗细则根据应用需要设定。在基于图像的交互 NPR 技术中,笔刷方向通过用户交互指定,一般交互量越大,所绘制的效果越接近手工绘画。在基于图像的自动 NPR 技术中,一般需要提取物体的边缘来提供笔刷方向信息,物体区域中的笔刷方向要通过对边缘信息进行某种运算来确定。为了强化手工绘画效果,可以采用多层处理技术进一步变化笔刷的长短和粗细。

在基于模型的三维自动 NPR 技术中根据笔刷长短有不同的处理方法。对于长笔刷一般先检测物体边缘及物体表面褶线,然后再进行笔刷绘制,对于短笔刷(包括非传统笔刷)则在物体网络内分布,其位置进行局部随机扰动。在物体宏观上,笔刷的分布密度可结合光照模型计算出来的明暗进行控制。在基于模型的三维交互 NPR 技术中,用户可以选择多种预先设计的笔刷纹理绘制到物体边缘和表面上。

NPR 技术种类繁多,绘制出的效果风格各异。基于这种特点,NPR 技术仍在继续发展中。

参考文献

1. Non-photorealistic rendering. Siggraph '99 Course 17, August, 1999
2. Robert D et al. WYSIWYG NPR: Drawing Strokes Directly on 3D Models. ACM Transactions on Graphics (TOG), 2002, 21(3): 755 ~ 762 (于金辉)

fenbu jiaohushi fangzhen

分布交互式仿真 (distributed interactive simulation, DIS)

在高速计算机网络支持下,将分布在不同地点的仿真系统集成起来,通过仿真实体间的实时数据交换,在时空一致的人机交互仿

真环境中并行进行仿真的一种先进仿真技术。

分布交互式仿真是并行仿真网络化思想的进一步发展。尽管在 20 世纪 80 年代初就出现了多台计算机联网仿真,并作为一种松连接的并行仿真技术加以使用(参见并行仿真),但是,随着计算机网络技术的迅速发展,网络传输速度大大提高,而多处理机系统无论其硬件技术还是其软件技术(特别是并行操作系统和并行语言)却步履缓慢,加上仿真技术所面对的问题规模日益庞大,种类也日趋增多,于是人们重新着力于分布式仿真技术的研究。

20 世纪 80 年代中期,美国国防高级研究计划局(DARPA)和陆军共同制定了 SIMLET 计划,其主要目的是将分散在各地的多个地面车辆仿真器(坦克、装甲车等)用计算机网络连接起来,进行组级的协同作战任务的仿真训练,这可以被认为是分布交互式仿真技术典型应用的开始。

在 SIMNET 实施成功的基础上,分布交互式仿真(DIS)的标准得到重视。1996 年美国国防部正式颁布了一个建模与仿真领域的通用技术框架。该框架由三部分组成,即:任务空间概念模型(CMMS)、高层体系结构(HLA)和一系列的数据标准。其中 HLA 是该技术框架的核心,被认为是新一代分布交互式仿真标准,2000 年 9 月被 IEEE 批准成为国际标准。

分布交互式仿真系统从结构上可以视为由仿真结点和计算机网络组成。仿真结点本身可以是一台独立仿真计算机,甚至是一个仿真系统。与单独仿真计算机不同,分布交互式仿真(DIS)的仿真结点不仅要完成本结点的仿真任务(如运动学、动力学模型的计算,人机交互,仿真图形或动画的生成等),还要负责将本结点的有关信息发送到其他结点;同时,它也必须按一定周期接收其他结点发送来的信息,并作为执行本结点任务时的输入或条件。一般说来,DIS 中的结点往往是任务各异的,这要根据仿真系统的任务分配而定。而且,也并不要求计算机的类型、操作系统类型、编程语言类型一定相同。为了保证这种异构系统的信息共享与通信,需要在应用层定义信息交换的格式和内容。这些交换的信息单元称为协议数据单元(PDU),使得交换的信息内容与通信硬件和软件无关,从而,易于采用通用的商业网络软硬件环境(如 TCP/IP 协议)。

从“分布交互式仿真系统”的名称就不难分析出此类并行仿真系统的特点,即分布性、交互性。

分布性包含功能及地理位置两方面的分布。地

理位置上的分布性通过网络实现其集成,一种典型的分布交互式仿真结构如图1。而功能分布性表现在:在分布交互式仿真系统中没有中央处理机,各仿真结点是平等的。每个仿真结点具有各自的资源,并负责对分配给它的仿真任务进行处理。分布

交互式仿真系统中结点一般具有自治性,即一个结点的加入与退出不会引起别的结点正常执行。实际上,大多数分布交互式仿真系统不是执行单一仿真任务的系统,往往是将功能各异的若干独立仿真器集成起来以完成更大规模的系统仿真。

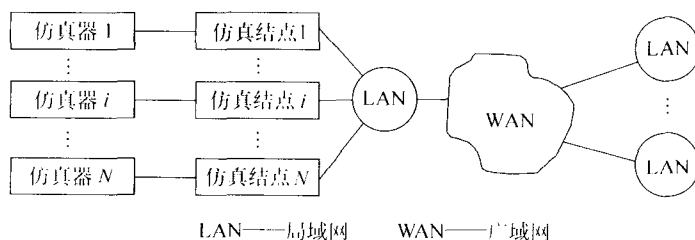


图1 典型的分布交互式仿真结构

分布交互式仿真(DIS)的交互性包括:①结点内部进程交互、人机交互;②结点之间的交互作用;③不仅人机交互及结点之间的交互要求实时性,而且要保证空间的一致性,称其为时空一致性。时空一致性是DIS的基本要求,也是DIS的本质特征。

参考文献

1. 柏彦奇. 联邦式作战仿真. 长沙: 国防科技大学出版社, 2001
2. 肖田元, 张燕云, 陈加标. 系统仿真导论. 北京: 清华大学出版社, 2000
3. 周彦, 戴剑伟. HLA 仿真程序设计. 北京: 电子工业出版社, 2002 (肖田元)

fenbushi caozuo xitong

分布式操作系统 (distributed operating system)

用于支持分布式处理系统的操作系统。和集中式操作系统相比,分布式操作系统在资源管理、进程通信和系统结构三方面有明显的区别。对于分布式操作系统而言,如果仍采用一类资源由一个管理者来管的集中管理方式,那么,当多个结点的用户要求使用某个结点上一类资源时,可能需要频繁通信,增加开销,所以分布式操作系统采用一类资源多个管理者的分布式管理方式。多个管理者相互协调,共同完成对资源的管理,以提高管理效率。分布式操作系统的进程通信有两种主要形式:消息传递和远程过程调用(RPC)。消息传递以进程间直接相互发送或接收消息实现,RPC则通过调用远程进程的过程实现消息的交换。在系统结构方面,分布式操作系统和集中式操作系统一样,也是由内核及

提供系统各种功能的模块组成。但在分布式系统中,每台计算机上都有一个内核,它实现对该计算机系统基本的控制,系统的其他模块则不均匀地分布在各台计算机上。这种分布可以是静态的,也可以是动态的,它不仅可节省系统开销,而且可以保证系统的稳健性。

分布式操作系统的机制实现可分为两类:面向进程模式和面向对象模式。面向进程的模式是通过一组进程来提供上述服务。通常,这些进程的规模较大,因此一个分布式操作系统中进程数目不多。进程间的同步及对用户进程的控制都是通过消息传递实现的。面向对象的分布式操作系统由一组对象组成,它们提供对系统的各种操作,这样的分布式操作系统由大量对象组成,每个对象负责在自己的局部环境中执行少量的操作。

采用面向对象方法、线程机制和微内核结构及客户-服务器工作模式是当前分布式操作系统研究和设计中的主要研究方向。

参考文献

1. 孙钟秀. 分布式计算机系统. 北京: 国防工业出版社, 1987
2. Fortier P J. Design of Distributed Operations Systems, Concepts and Technology. New York: Intertext Pub. and McGraw-Hill, 1986 (杜兴)

fenbushi chengxu sheji

分布式程序设计 (distributed programming)

适用于分布式处理系统的一种程序设计。它出现于20世纪70年代后期。

采用分布式程序设计时,一个程序由分布于系统各结点上的程序模块组成。这些模块可以同时执行,它们通过通信互相协调和交换数据。与经典的程序设计相比,分布式程序设计有三个特点:分布、通信和坚固性。

分布是指程序分为若干个可独立执行的模块,它们分布于系统的不同结点上。程序的分布可分为静态和动态的两种。静态分布是指在装入程序时就将组成程序的模块分别装入不同的结点,在程序执行时不再增加独立执行的模块。动态分布是指在程序执行的过程中产生可独立执行的模块。一个模块在执行中可以撤销它所创建的模块,在它自己被撤销时,一般说,也应将它创建的模块撤销。

通信是指各程序模块之间传递消息。由于分布于各个结点上的程序模块是相互关联的,它们在执行中需要交换数据和互相协调,因此通信是必不可少的。在分布式程序设计中相互通信的两个模块往往是位于两个不同的结点上,通信必须通过连接各个结点的网络进行。这种通信的实现比位于同一台计算机内的两个进程之间的通信的实现要复杂得多。

坚固性是指当系统的某些结点失效时系统仍能维持工作的能力。由于系统是由多个结点组成,从硬件角度看它可能具有坚固的性能。但是要真正具有坚固性还必须依靠软件。当出现结点失效的故障时系统除作一些信息保护外是无能为力的。具体的维护工作只有用户程序的编制者才知道。因此分布式程序设计语言往往要给用户提供一些手段来处理结点失效。

采用分布式程序设计来处理问题时必须提供分布式程序设计语言。分布式程序设计语言必须提供程序分布、程序通信和故障处理三种功能。设计一种分布式程序设计语言有两种方式:在某种程序设计语言上增加分布和通信等功能或重新设计一种新的语言。

参考文献

孙钟秀. 分布式计算机系统. 北京: 国防工业出版社, 1987

(孙钟秀)

fenbushi chuli xitong

分布式处理系统 (distributed processing system) 将不同地点的或具有不同功能的或拥有不同数据的多台计算机用通信网络连接起来,在控制系统的统一管理控制下,协调地完成信息处理任务的计算机系统。分布式处理系统是**计算机网络**

在功能上的延伸,分布式处理系统中的多台计算机除了可以相互通信和共享资源外,还能协同工作。分布式处理和并行处理的界限对某些系统来讲是模糊的,例如工作站簇既是分布式处理系统,又可看成是松散耦合的并行处理系统。一般认为,集中在同一个机柜内或同一个地点的紧密耦合多处理机系统或**大规模并行处理系统**是并行处理系统,而用**局域网**或**广域网**连接的计算机系统是分布式处理系统。松散耦合并行计算机中的并行操作系统有时也称为分布式操作系统。

分布式处理系统可以有多种组成方式,可由机型和功能大致相当的计算机按水平方式组成,也可由主机系统和工作站、个人计算机、终端等按垂直方式组成。**客户-服务器计算系统**、**计算机簇**、**异构型计算机系统**(参见**分布式异构型计算机系统**)、**计算机支持的协同工作系统**等都是分布式处理系统的例子。

分布式处理系统包含硬件、控制系统、接口系统、数据、应用程序和人6个要素。

硬件包括计算机本身的硬件装置,例如处理器、存储器、输入输出设备等,还包括各种网络通信设备,例如各种局域网、广域网、**网络互连设备**、交换机以及各种传输媒体等。

控制系统是分布在各种硬件装置中的控制程序的集合。控制系统主要包括:①**分布式操作系统**,例如 Mach 操作系统;②**分布式数据库**(参见**分布式数据库系统**)和**分布式资源管理工具**,例如 Sybase 数据库;③**通信协议**(参见**网络协议**)和**网络管理协议**,例如 OSI 协议和 **TCP/IP 协议集**;④**不同媒体的输入输出控制程序**等。

接口系统包括两个部分,即应用程序开发人员接口与一般用户接口,应用程序开发人员接口有时也称为群体开发平台,包括各种语言的库函数和开发工具,还具有对各种不同应用命令的分布协调处理、信息共享与互斥的控制、不同用户和不同媒体的识别以及多媒体信息输入输出处理等功能。一般用户接口包括多种操作命令和图形界面,除了 Windows, Motif 以及 Openlook 等窗口系统提供友好的图形界面外,还可提供声音输入、手写体输入以及电视会议系统的实时多媒体输入等的方法。

数据指存储在分布式数据库中的各种数据。

应用程序包括的范围非常广泛,常用的有**文件传送系统**、**电子邮件系统**、**电子数据交换系统**、**计算机集成制造系统**等。另外,分布式会议系统、远程教育系统、分布式科学工程计算系统、分布式 CASE 以

及远程医疗监护系统等都是分布式系统的应用例子。

人是分布式处理系统的使用者,设计分布式处理系统时必须考虑使用者所具有的能力和知识,特别是在群体计算系统中,系统设计人员还必须考虑使用者的文化习惯和所处的社会结构等因素。

分布式处理系统的典型物理结构如图 1 所示。在图中所示的分布式处理系统中,用户通过工作站、个人计算机以及终端等透明地使用分布式系统的各种资源。这些资源包括服务器、大型数据库系统和具有大规模计算能力的超级计算机等。用户只需向系统提交任务,系统根据负载情况自动调配各种资源去完成用户提交的任务,用户不必了解系统完成任务的过程和具体细节。分布式处理系统的软件结构如图 2。

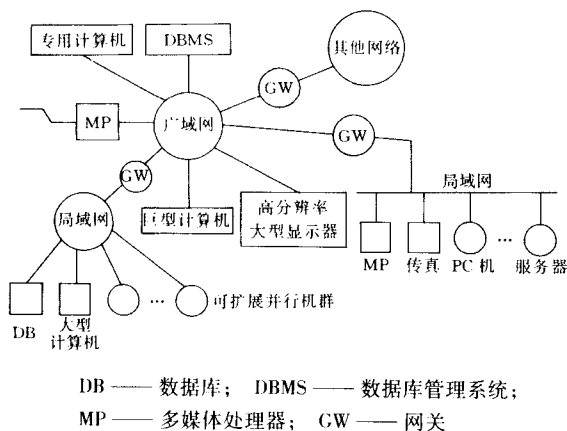


图 1 分布式处理系统的典型物理结构

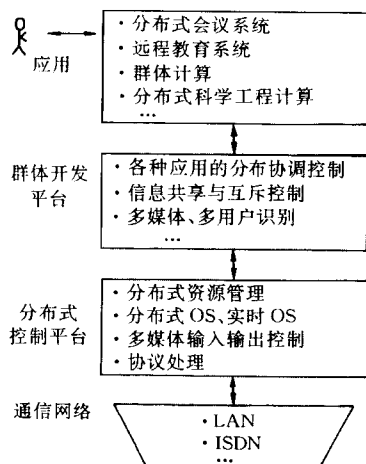


图 2 分布式处理系统的软件结构

图 3 给出了从系统管理人员的角度所看到的分布式处理系统的逻辑结构。用户通过工作站、个人计算机等的控制程序与外部系统连接,而不必知道外部资源的确切位置。

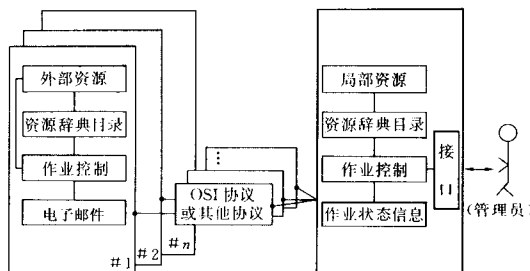


图 3 分布式处理系统的逻辑结构

随着网络技术、分布式操作系统和分布式数据库等的发展,分布式处理系统已越来越成熟,从而也得到了越来越广泛的应用。(张尧学)

fenbushi duomeiti xitong

分布式多媒体系统 (distributed multimedia system) 集成了通信、计算和信息处理的分布式系统。它能处理、传送、管理和展现多媒体信息并具有服务质量 (QoS) 的保证。

分布式多媒体系统的主要特点有: ①技术集成 它集成了信息、通信和计算系统,形成一个统一的数字处理环境; ②多媒体集成 在这个集成环境中不仅要处理离散的(与时间无关的)媒体数据,还要处理连续的(与时间有关的)媒体数据; ③实时性能 要求其存储系统、处理系统和传输系统都具有实时性能,因此,海量存储、高输入输出速率、高网络带宽和高 CPU 处理能力是必需的; ④系统级的服务质量保证 支持沿着从发送者,经传输网络,到接收者的整个数据路径的端到端的不同的服务质量需求; ⑤交互性 支持用户和系统之间的双向通信,并允许每个用户都能控制所访问的信息; ⑥多媒体同步支持 保持单一连续媒体流内媒体帧的连续播放和相关的对象之间的时序关系; ⑦支持标准化 在信息内容、展现格式、用户界面、网络协议和信息设备等异构的情况下,能够实现互操作。

分布式多媒体系统的支撑技术分为 3 个方面: ①信息子系统(存储方面) 包括多媒体服务器、信息建档、多媒体数据库系统,支持多媒体信息的储存与检索,能一致、安全、可靠地管理数据,响应大量并

发的用户请求并保证服务质量。②通信子系统(传输方面) 包括传送媒体和传输协议,它将用户与分散的多媒体数据源连接起来,按不同的服务质量传送多媒体数据,如实时传送视频、音频数据,无差错地传送文本数据。③计算子系统(处理方面) 包括多媒体计算机、操作系统、展现和著作工具,以及多媒体管理软件;用户通过计算子系统管理多媒体数据。

分布式多媒体系统的应用可以划分为3类典型的应用系统:①**流媒体** 在此类系统中,用户可随时访问媒体数据源,例如交互式视频游戏、新闻点播、视频节目点播等;②**远程协作** 此类系统支持多个用户克服时空阻隔共同参与完成一项任务,例如远程教学、多媒体电子邮件、电视会议、桌面会议、远程医疗、协同编著等;③**超媒体** 此类系统提供超媒体文档的浏览,例如数字图书馆、电子百科全书、多媒体杂志、多媒体文档、多媒体信息亭、多媒体课件和万维网(WWW)浏览等。(徐光佑)

fenbushi gongxiang cunchu

分布式共享存储 (distributed shared memory) 参见共享存储。

fenbushi jisuan huanjing

分布式计算环境 (distributed computing environment) 在具有多地址空间的多计算机系统上进行计算和信息处理的软件环境。过去,分布式计算环境是指为分布式计算机系统提供的软件环境。后来,这一概念被扩展为包括各种分布式存储的并行计算机系统提供的计算环境。这些系统的主要特征是多个用户进程在多个通过网络互联的计算机结点上运行,每一进程有自己的地址空间,进程之间通过消息传递模式进行通信。分布式计算环境为这些分布计算提供各种服务和工具,以实现资源共享、并行计算和高可用性。

分布式系统或并行系统可由不同类型的计算机结点组成,这些结点可实现资源共享。假设一个系统包括几个功能强的结点和数百个功能弱的结点,功能强的结点有强大的计算、存储能力和丰富的软件和信息资源,但价格昂贵;功能弱的结点价格便宜。分布式计算环境能让弱结点上的用户利用强结点上的服务,分享强结点上的各种资源。

通过并行计算技术,分布式计算环境可以集合

多个结点上的处理器、内存和硬磁盘资源,用于解决大问题,提高系统吞吐率,减小响应时间。比如某一应用程序需要2 GB的内存和1 a的计算时间,通过数据分割、工作分割等并行技术,此程序可以被分布到16个结点上。在理想情况下,每一结点只需有128 MB的内存,整个程序可在不到一个月的时间内算完。分布式计算环境可为这样的并行应用提供程序设计、通信、任务管理和并行输入输出等的工具。

并行分布式系统应具有很好的可用性,即整个系统不应因为部分结点出故障而崩溃。一个理想的分布式计算环境应该提供一定的故障处理能力,例如故障检测、自动备份、将故障通知系统管理员和用户、重构系统让应用程序绕过故障结点,将程序从故障结点迁移到正常结点等。

下面简述3个分布式计算环境,它们是:DCE、PVM和MPI。

分布式计算环境DCE是由开放软件基金会(OSF)于1992年发布的分布式计算环境标准,已被广泛采用。DCE包括如下关键概念和技术。

(1) **线程** DCE支持在同一进程的地址空间内创建和管理多个线程,每个线程是一控制流序列。线程比进程开销小。多线程使1个服务器进程能同时为多个客户服务。

(2) **远程过程调用(RPC)** RPC允许一个结点上的应用程序能调用另一结点上过程,就像调用本地过程那样方便。

(3) **名录服务** 名录服务存放并管理系统中各种资源的名称和属性等信息,以便任一结点上的任一进程查找。

(4) **分布式文件系统** 分布式文件系统使用户可以访问任一结点上的文件,就像访问本地结点上的文件那么方便。分布式文件系统也应具有一定的容错功能。

DCE的其他关键技术还有:安全服务可提供安全通信和安全资源访问,分布式时钟服务可将各结点的局部时钟同步,无盘服务可使无硬磁盘的结点能使用其他结点的硬磁盘。

并行虚拟机PVM是美国的一些大学和国家实验室在20世纪90年代初开发出来的公开的分布式计算环境软件,已在世界范围内被广泛采用。PVM通用性强,所需资源较少。将多个计算机用任何网络连接起来,再装上PVM,即成为一个分布式计算机系统。该系统的结点可用不同的平台,从微型计

算机到超级计算机均可,因为 PVM 支持异构系统。PVM 提供的服务主要有虚拟机管理、进程管理和消息传递。

消息传递接口 MPI 是由欧美的 40 多个国家实验室、大学和计算机厂商共同制订的一个公开的消息传递标准。1994 年发布了第一版,1995 年开发了可移植于多种平台的 MPI——MPICH,1996 年发布了第二版 MPI-2 的草稿。由于其公开性、标准性、可移植性和高性能消息传递功能,MPI 已被广泛采用。MPI 比 PVM 具有更强的消息传递功能。MPI-2 还增加了进程管理、单边通信和并行输入输出等功能。

另外,值得重视的还有万维网(WWW),由于采用公开的、标准的、与平台无关的超文本描述语言以及免费的浏览器软件,它提供了方便、友好的界面,使用户通过因特网能查询和传递各种类型的信息(文本、图像、声音、动画等)。Sun Microsystems 公司于 1995 年推出的 Java 语言可让用户通过因特网执行各种分布式程序。有人认为,今后的主流分布式计算环境将是基于万维网的、与平台无关的窗口系统。

(徐志伟)

fenbushi ruanjian xitong

分布式软件系统 (distributed software system) 支持分布式处理的软件系统。它包括**分布式操作系统**、**分布式程序设计语言**及其编译(解释)系统、**分布式文件系统**和**分布式数据库系统**等。

分布式操作系统负责管理分布式处理系统资源和控制分布式程序运行。它和集中式操作系统的区别在于资源管理、进程通信和系统结构等方面。

分布式程序设计语言用于编写运行于分布式计算机系统上的分布式程序。一个分布式程序由若干个可以独立执行的程序模块组成,它们分布于一个分布式处理系统的多台计算机上被同时执行。它与集中式的程序设计语言相比有三个特点:分布性、通信性和稳健性。

分布式文件系统具有执行远程文件存取的能力,并以透明方式对分布在网络上的文件进行管理和存取。

分布式数据库系统由分布于多个计算机结点上的若干子数据库系统组成,它提供有效的存取手段来操纵这些结点上的子数据库。分布式数据库在使用上可视为一个完整的数据库,而实际上它是分布在地理分散的各个结点上。当然,分布在各个结点上的子数据库在逻辑上是相关的。

发展概述

分布式软件系统的发展已有十几年的历史,它的发展大体上已经历了两个阶段。从 20 世纪 70 年代中期到 80 年代初期是开创阶段。这一阶段的主要目标是发展分布式系统软件来支持实验性的分布式硬件系统以成为一个分布式处理系统。例如,支持通信和资源共享的网络操作系统 RSEXEC, NEW (1978) 和 XNOS (1978), 分布式操作系统 ACCENT / NATH-1 (1981), ZCZOS (1982), ARGUS (1983) 等;在这段时间内,提出了十余种分布式程序设计语言的建议,如分布式进程, CSP, * MOD, ARGUS, CSM 以及 DMODULA 等。此外,在同步算法、通信机制等方面也进行了研究。这一阶段的研究是理论性的或应用基础性的,还没有实用的系统。

随着应用的迫切需求,80 年代中后期在开创性研究的基础上很快就进入了边研究边开发应用的第二阶段。密切结合应用来进行研究是这一阶段的特点。正如著名的计算机科学家霍尔(C. A. R. Hoare)所指出的那样:“分布式计算是一个迷人的课题,具有很大的理论和实践意义。它的迷人之处在于理论和实践同时发展以使得实践推动理论,理论又指导实践。”在这一阶段开发了一些具有实用价值的分布式软件。如分布式操作系统 V-KERNEL (1984), EDEN (1985), AMOEBA (1987), 分布式程序设计语言分布式 C 等, 分布式文件系统 CFS, AFS 和 NFS 以及通信机制 RPC (远程过程调用) 等。有一些研究成果,如实时系统的调度算法、并行 FFT 算法等也开始应用于实际系统。

20 世纪 90 年代以来,分布式软件技术的发展开始进入第三阶段。这一阶段的特点是除了继续理论研究外,着重发展实用化技术、应用技术及商品化技术。从研究工作上看,它将和其他计算机新技术相结合,特别是和并行处理技术密切结合而成为当前计算机技术中最重要、最活跃的新技术之一;从应用方面看,分布式处理技术将更广泛地渗透到实用系统中。

基本内容

分布式软件系统研究的主要内容有:分布式操作系统,分布式程序设计语言,分布式文件系统和分布式数据库系统等。

分布式操作系统的研究内容包括:通信机制(如消息传递和远程过程调用(RPC),通信协议的规约和验证);命令和保护(如命令和映射,名字服务器);资源管理(如进程分配、调度和存取控制,负

载平衡和分布式死锁控制);容错(如冗余技术);服务(如实用程序);异构性和透明性;安全性(如安全策略,保护机制)等。

分布式程序设计语言的研究内容包括:并发程序设计模型;并发性的形式化处理;并发和分布式程序设计语言的设计与实现等。

分布式文件系统研究的主要内容包括:高性能的远程存取;高效性;信息保护;系统可伸缩性和共享数据的并发存取等。

分布式文件系统可分为三类:远程磁盘系统,实现共享盘的存取;块级存取系统,实现指名远程文件的块的存取;文件级存取系统,实现共享文件的存取。

分布式数据库系统的研究内容包括:分布式数据库的设计;分布式查询处理;分布式目录字典管理;分布式并发控制;分布式死锁管理;分布式数据库管理系统的可靠性;操作系统的支持;异构型分布式数据库等。

自从1976年由美国CCA公司开发了第一个实验性分布式数据库管理系统SDD-1以后,相继开发了比较著名的系统有R*,POREL,SIRIUS-DELTA,MULTIBASE等;同时,商品化的分布式数据库管理系统也陆续产生,如:ENCOMPASS,ORACLE,SYBASE等。

展 望

未来十年内,分布式软件系统将会获得更快的发展,在一些重大应用领域,例如,大规模并行处理系统、信息高速公路、空间开发等方面将起重要作用。分布式软件系统将向实用化、开放式方向发展,并预计下列技术会获得更快的发展。

(1) 并行分布式处理软件 对一个高度并行和分布控制的大规模并行处理系统而言,亟待发展有效的分布并行软件系统,如并行操作系统,并行分布程序设计,并行编译等。

(2) 实时分布式软件 许多实时系统都是分布式处理系统,它不仅要求处理结果正确,而且要求结果在规定时间内产生。因此,需要发展实时分布式软件。今后的主要研究内容包括:实时分布调度、同步通信和容错等。

(3) 分布和并行数据库系统 分布和并行数据库系统是分布式处理系统实用化的一项关键性工作,至今,在分布数据库系统领域已取得了若干成果,但实用性方面突破性进展不多,而并行数据库系统的研究还刚起步。

(4) 开放分布式处理 开放分布式处理是分布式处理技术发展的高级阶段,它是试图解决网络环境下分布式软件系统接口问题的新兴技术,其目标是在网络的集成系统中,提供给应用程序一个一致的接口模型,以实现分布透明性、互操作性和易移植性。

参考文献

1. Ananda A L and Srinivasan B. Distributed Computing Systems: Concepts & Structures. IEEE Computer Society Press, 1991

2. 谢立,孙钟秀. 分布式数据处理. 北京:国防工业出版社,1990

3. 孙钟秀. 分布式计算系统. 北京:国防工业出版社,1985 (谢立)

fenbushi shujuku

分布式数据库 (distributed database) 参见分布式数据库系统。

fenbushi shujuku xitong

分布式数据库系统 (distributed database system) 由分布于若干个计算机结点上的若干子数据库系统所组成的数据库系统,它提供有效的存取手段来操纵这些子数据库。分布式数据库在使用上可视为一个完整的数据库,而实际它是分布在地理上分散的各个结点上。分布在各个结点上的子数据库在逻辑上是相关的。操纵这些子数据库的软件称为分布式数据库管理系统 (DDBMS)。

分布式数据库系统在集中式数据库系统的组成基础上增加了三个部分:分布式数据库管理系统 DDBMS(又称网络数据库管理系统 NDBMS),网络数据字典 NDD 和网络存取进程 NAP。

分布式数据库管理系统的主要功能如下:

(1) 接收用户请求,并判定送往何处,或必须访问哪些结点计算机才能满足该请求。

(2) 访问网络数据字典,或者至少了解如何请求和使用其中的信息。

(3) 如果目标数据存在于系统的多个结点计算机上,就可以进行并行处理。

(4) 在用户进程、局部数据库管理系统和其他计算机的数据库管理系统之间进行通信服务。

(5) 在一个异构型分布式处理环境中,还需要提供数据和进程移植的支持。所谓异构型是指在系

统的每个结点上的数据库系统是有差别的。

网络数据字典为分布式数据库管理系统提供达到其功能的数据单元位置的必要信息。网络存取进程提供一个结点的分布式数据库管理系统的进程和通信子系统之间的接口。它使用高级协议来影响内部结点的分布式数据库之间的通信。

从结构上分,分布式数据库系统可分为两类:全局型分布式数据库系统和联邦型分布式数据库系统。

全局型分布式数据库系统是最常用的类型。在这种结构中,一个数据库以完整的方式分布在一个计算机网络的各个结点上。结点间的所有信息交换都由系统软件处理,它对用户是透明的。这样结构具有如下特征:采用完全分布方法;允许有多份副本;由分布式数据库管理系统来控制进程间的同步;系统进行故障恢复处理;系统具有较高性能。

采用这样结构的分布式数据库系统中,用户看到的是一个完整的数据库,他们不必了解他们的数据所在位置。同样,对于应用程序而言,通信将在数据库管理系统中进行,并且由分布式数据库管理系统来处理。

联邦型分布式数据库系统的结构用于以下情况:存在若干个现存的、分离的数据库结点,且这些数据库的类型可以相同,也可以不同。这类结构具有如下特征:分离的数据库必须在逻辑上以某种方式联成一个整体;从单个终端用户的观点来看,他们要求访问若干不同的数据库,但是,不必了解新的接口进程。用户所需要的是一个“友善的”和与所有查询相容的接口。同时,事务处理由系统软件自动进行。

联邦型结构往往在“更新”情况下采用,因此受到原有软件的限制。另外,由于开销原因,也不宜有网络范围内的“恢复”机制。最后,尽管有 NDD 和合作的数据字典提供有用的帮助,但用户通常仍需知道它们数据的存放位置。

分布式数据库系统是在集中式数据库系统和计算机网络的基础上发展起来的,它是分布式数据处理的关键技术之一,在办公自动化、管理信息系统等方面有着广泛的应用。例如,银行管理系统,企业管理系统,军事指挥训练系统,航空公司或旅馆的预订系统等。

参考文献

谢立,孙钟秀. 分布式数据处理. 北京:国防工业出版社,1990
(郑宇华)

fenbushi wangluo guanli

分布式网络管理 (distributed network management) 把网络管理的各个要素分布在整个网络上的一种计算机网络管理。网络的公共管理信息要素主要有网络管理功能、网络管理信息等(参见 OSI 管理体系结构)。网络的公共管理信息要素的分布方式有以下 3 种:

(1) 重复 使一个要素同时存在于网络的不同系统,目的是为了提高整个系统的可靠性;

(2) 分散 把某个要素分割成若干较小的要素,然后把它们分散在网络上,目的是为了使较小的系统能完成网络管理;

(3) 合作 由网络上多个管理系统相互合作,共同完成一个网络管理功能,目的是为了完成复杂的网络管理功能。

分布式网络管理的优点是可靠,同时,由于访问的集中性(即对特定的网络管理功能或管理信息,往往在比较固定的系统上对它们进行访问),在设计分布式网络管理时,可以安排得比较合理,从而提高整个网络管理的效率。它的缺点是如果分布不合理,可能造成管理效率低下,甚至影响整个网络的有效运行。

参考文献

Subramanian M. Network Management: Principles and Practice. Addison - Wesley, 1999
(钱松荣)

fenbushi yigouxing jisuanji xitong

分布式异构型计算机系统 (distributed heterogeneous computer system) 由多个不同种类的计算机平台或应用子系统通过网络连接而成的计算机系统。计算平台(简称平台)是指计算机的硬件系统和操作系统的组合。应用子系统是指在平台之上支持某一特定应用的软件,例如数据库。

下面通过几个例子来说明异构性。例如,某一分布式系统由 3 台微型计算机通过以太网连接而成。所有微型计算机都采用奔腾 Pro 处理器芯片和与 IBM 微型计算机兼容的硬件体系结构,但用了 3 种不同的操作系统:1 台用 Windows 95,1 台用 Windows NT,1 台用 Linux。这就是一个异构系统,因为有 3 种不同的平台。再例如,某一分布式系统在它的所有计算机中都采用同一操作系统 Windows NT,但计算机采用了不同的处理器:奔腾和 Power PC,这也是一个异构系统,因为用了两种不同的平台。一

个分布式系统的所有平台即使都相同,但如果采用了不同的数据库(例如 Oracle 和 Sybase),该系统也是异构的。

绝大多数分布式计算机系统都是异构的。主要原因有 3 个:第一,用户对计算机系统的需求是异构的。比如用户需要高性能的计算能力、高输入输出吞吐量的文件服务器和快速的三维图形终端,以及强有力的应用软件。满足这些需求只能采用不同的平台。第二,异构性可以提高系统的性能价格比。例如某一系统可由 1 台昂贵而性能高的服务器和 10 台便宜的微型计算机组成。微型计算机用户可以通过网络共享服务器的资源。这个异构系统比由 11 台服务器组成的同构系统价格低廉得多。第三,用户经常依靠连接多个子系统的办法来扩大自身的系统,这样也常常导致异构系统。

异构系统所要解决的最关键的问题是互操作性,即允许在多种平台间的数据通信和程序执行。互操作性可用 3 种技术来实现:第一是采用标准的接口,例如 Internet 上的几百万台计算机都可以用 TCP/IP 协议集通信。第二是用中间件,即用来支持平台间相互作用的软件,它支持平台间的数据通信和程序执行。第三是用与平台无关的编程工具,它开发出来的程序可以在不同平台上执行。这样的编程工具包括 Java 语言。Java 的字符数据类型采用 16 位的国际标准码,包括世界上很多语言。而 C 语言采用 8 位的 ASCII 码,只适用于西文,不足以表述有 5 万多个字符的中文。Java 语言不允许直接调用操作系统的过程,必须通过所规定的接口,这样保证了与平台的无关性。

(徐志伟)

fenbushi zhuanjia xitong

分布式专家系统 (distributed expert system, DES) 逻辑上或物理上分布在不同的物理结点上的若干专家系统协同求解问题的系统,是人工智能领域中分布式人工智能分支的一部分。

在现实生活中,有很多复杂的问题,涉及多学科、多领域的知识,人们在求解时需要一个群体(多个专家)来协同求解该问题,以求取得理想的或有效的结果,或者快速取得结果。因此,需要构造多个专家系统协同求解问题的系统。同样,反映在用单个专家系统求解问题时,其领域知识的局限性和单一性,致使系统十分脆弱,也有多个专家系统协同求解的需要。例如,疑难病会诊,需要通过多个领域的专家系统协同求解,以增加求解结果的有效性;工业

生产中多个机器人控制,企业管理中的决策,需要通过不同领域知识的专家系统协同求解,以改善求解的整体性能。

群体协同求解是把一个复杂问题分解为多个简单问题求解,便于并行执行,或可动态地组织求解结构,增加求解问题的速度和灵活性。

分布式专家系统中每个结点上的专家系统(简称结点专家系统)与单个专家系统是不同的,它不仅应具有求解特定问题的能力,而且还应具有以下性能:

(1) 每一结点专家系统仅有有限知识,包括领域的有限知识,群体任务规划调度的有限知识,预测其他结点专家系统能力的有限知识等;

(2) 要有分解复杂问题为若干子问题,并根据逻辑关系进行优先顺序排队的能力;

(3) 要有选择合适的专家系统求解子问题的能力;

(4) 要有与其他结点专家系统合作与通信的能力。

分布式专家系统在求解问题的类型和方法上不同于一般分布式处理系统,它是在环境不确定性(不具有确定数量的结点专家系统)、数据不确定性(不具有完全的一致性的局部数据)、控制不确定性(不具有另一结点专家系统完全精确的活动模型)的情况下求解问题。因此,要求结点专家系统在环境变化时有修改自身的行为和规划以及与其他结点专家系统合作、通信的策略。

分布式专家系统在分布式网络的风格上也不同于一般分布式处理系统。典型的分布式处理系统具有多个独立地并发执行的任务,为共享物理或信息资源而进行任务间的隐式交互活动。而分布式专家系统却显式地知道每个结点专家系统的功能及分布位置,在此基础上作出合作与通信的决策,相互地协同求解问题。

关于分布式专家系统的实现有许多不同的构思,一种典型的结点专家系统体系结构如图 1 所示。主要组成部分的功能描述如下:静态规划器根据用户提交本结点专家系统的任务,用某种分解方案把任务分解为多个子任务,并以一定层次结构存放于子任务队列中。调度器从子任务队列中获取任务,当该任务可由本结点求解器独立完成时,传送给求解器,否则传送给协同器。调度器还接受协同器传送来的需本结点专家系统完成的任务,根据规划原则插入相应队列,等待再分配执行。协同器管理本

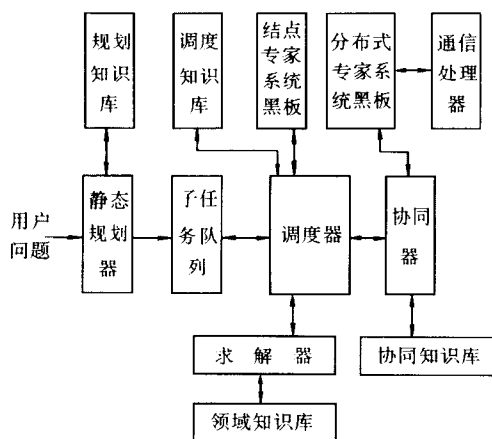


图1 结点专家系统体系结构

结点专家系统与其他结点专家系统的交往,确定合适的能完成任务的结点专家系统。如不能确定,则可采用广播征询方式选择,或者接收其他结点专家系统传送来的任务或回答对方的征询。求解器和领域知识库起着与单个专家系统相同的作用。

为使整个分布式专家系统达到求解问题的全局合理性和有效性,其关键是解决多个专家系统之间的协作策略。协作策略由启发式规则组成。这些规则可分为两类,一类为组织策略规则集,一类为信息策略规则集,分述如下。

组织策略决定:①如何把较大任务分解为若干子任务的方案。一般分解成层次结构,子任务向父任务报告结果。②如何预测和选择可完成任务的专家系统及与其通信的路径,并给予共享资源的许可,而对其他结点专家系统则给予限制。合理的通信和资源共享能使系统有效地工作。③结点专家系统之间的行为采取什么指导原则,是自我利益为主的竞争性协作,是协商性协作,还是精确功能划分性协作等。组织策略的确定是协同求解问题的前提。

信息策略作出结点专家系统之间如何通信的决策。组织策略已决定了协作者之间通信的网络结构,信息策略仅作出何时通信和如何通信这样大量的低级决策。诸如广播方式通信还是选择单个对象通信,主动通信还是请求通信,单次传输还是多次传输,确认通信还是非确认通信,通信内容的时限等。信息策略依赖结点专家系统的分布模型和通信条件以及任务的特性。

在设计一个分布式专家系统时,必须考虑下述几个基本问题。

(1) 求解问题的描述、分解和分布 问题的描述是指对一类问题的特性、属性及其可能的求解方式进行抽象,产生出包括问题求解环境、可能的求解方法等信息。为问题在系统中的分解与分布、子问题间交互提供依据,并用来指导设计者研究专家系统间的协调,以求设计出有效的分布式专家系统。

问题的分解不仅要考虑问题的特点,而且要将子问题求解要求与各结点专家系统的能力和资源相匹配。针对不同的问题与系统类型,可以有不同的分解标准,减少控制或数据依赖是大多数分布式专家系统中问题分解的标准。对于强调可靠或需要多种方法求解问题的分布式专家系统,分解的子问题之间需要有一定冗余,而在通信资源有限的分布式专家系统中,则要求尽量粗粒度化地分解子问题,以减少通信与协调的开销。

子问题分布可以在问题求解之前由系统设计者完成,也可在系统运行过程中动态地进行,以适时地平衡负载和避免瓶颈的出现。

(2) 结点专家系统间的交互通信 问题分解与分布确定了结点专家之间的相互依赖,这就要求设计它们间的关系模式。典型的模式有:协商、市场交易、合同、主从等关系。另外还需设计合适的通信协议与语言来实现通信。

(3) 建立合理的行为指导原则 分布式专家系统研究的关键是如何在结点专家系统没有全局知识的情况下,使整个问题求解表现出全局一致的行为,即能高效地求出表达清晰的有效解。这就需要设计合适的指导原则来指导各个结点专家系统的问题求解。例如,规划结点专家系统的问题求解,以避免冲突、减少冗余、增加协作;增加结点专家系统的全局知识,提高通信的相关性、及时性和合理地利用资源;采用自适应的组织结构与算法,使分布式专家系统能够适应变化的外界环境等。

(4) 建立和利用其他结点专家系统的模型 在分布式专家系统中,一个结点专家系统为了了解另一个结点专家系统,需要建立后者的模型。根据此模型,一方面可以推断出该结点专家系统的知识范围、信念、行为与目标,评价来自其他系统的数据,修改或选择合适的组织策略和通信协议,增加灵活性;另一方面,可检测结点专家系统之间的一致性、不兼容性、不互识性等矛盾。消除矛盾的方法有:标准化、权威专家仲裁与调停、参照经验与先例、回溯检查并修正基本假设等。这种模型一般应包括:对该结点专家系统的资源、解题能力、组织策略和通信

策略的知识。

(5) 工程实现 工程实现分布式专家系统的关键是选择合适的开发工具。目前,常用的实现工具是基于对象的并发程序设计和黑板结构。

分布式专家系统是 80 年代发展起来的一种专家系统与分布式处理相结合的研究领域。其基本理论、体系结构、协作策略、开发工具等正在发展之中,尚未成熟,但已出现了不少实用的系统。分布式专家系统作为专家系统领域中的一个重要发展方向,已显示出它的实用价值。

参考文献

Huhns M N ed. Distributed Artificial Intelligence. Los Altos, California: Morgan Kaufmann.

(施鸿宝 胡蓬)

fenpian

分片 (fragmentation) 把逻辑上完整的数据库分成若干部分的一种分布式数据库技术。在分片设计中,每个全局关系被分成若干个不相重叠的部分,称为片。在全局关系和片之间的映射由片模式定义,这种映射是 1 对多的关系。通过分片方法获得的片是全局关系的一个逻辑部分,称为逻辑片,一个逻辑片储存到一个结点时,称为物理片。一个逻辑片可储存到多个结点,因此,一个逻辑片可以对应多个物理片。

存在两种基本的分片方法:水平分片和垂直分片。水平分片的主要思想是把一个全局关系的元组划分成多个子集,每个子集包含有公共地域信息的数据;垂直分片是把全局关系的属性按组划分为若干子部分。基于上述方法生成的片有以下 4 种类型:

(1) 原始水平片 原始水平片是在一个全局关系上执行一个选取操作的结果。例如,一个全局关系

供应站 (供应站号,供应站名,城市名)

原始水平片供应站 1,供应站 2 分别定义如下:

供应站 1 = Select_{城市名 = 北京} 供应站

供应站 2 = Select_{城市名 = 上海} 供应站

(2) 推导水平片 在某些情况下,一个关系的水平片不能基于它自己属性的一个性质,但可以从另一个关系的原始水平片中推导而得。例如,一个全局关系

供应 (供应站号,部件名,单位名,数量)

当要求根据供应关系来获得包含给定城市的供应站

的元组的片时,由于城市名不是关系供应的一个属性,所以它不能是原始水平片。但是,它是关系供应站的属性,所以可以成为一个推导水平片供应 1。其定义如下:

供应 1 = 供应 Semijoin_{供应站号 = 供应站号} 供应站 1

(3) 垂直片 垂直片是在一个全局关系上执行一个投影操作的结果。例如,一个全局关系

雇员 (雇员号,雇员名,工资,税金,单位名)

的一个垂直片雇员 1 的定义可为:

雇员 1 = Project_{雇员号,雇员名,单位名} 雇员

(4) 混合片 混合片是分别应用水平分片和垂直分片获得的结果。例如,一个混合片雇员 A 定义如下:

雇员 A = Select_{单位名 = 人事科} 雇员 1

为了保证分片结果的正确,根据分布方法对一个全局关系所定义的片必须满足如下规则:

(1) 正确性条件 全局关系的所有数据必须映射到所定义的某个片中,即不存在一个全局关系的数据项不属于所定义的任一片。

(2) 重结构条件 从定义的所有片可重新构造原来的一个全局关系。

(3) 不相交条件 所有定义的片是不相交的(垂直片中的主键码一般例外)。(杜兴)

fenshi chuli

分时处理 (time-sharing processing) 多个用户可同时与计算机系统交互处理作业的方式。

提供分时处理功能的操作系统称**分时操作系统**,它支持多个用户在各自的终端同时使用一台计算机而每个用户都感到好像自己各有一台独享的计算机。

分时是指把处理器的时间分成**时间片**,按时间片轮流的方式把处理器分派给各联机作业使用。如果一个作业在分得的时间片内不能完成计算,则暂时中断它,把处理器让给另一个作业使用,等待下轮再继续运行。由于现代计算机运行速度很快,用户作业运行轮转也就很快,实现了多用户共享一台计算机算题的方式。

分时处理的主要特点是:

(1) 同时性 若干个终端用户可同时使用计算机;

(2) 独立性 用户彼此独立,互不干扰;

(3) 及时性 用户的请求能在满意的时间内得到响应;

(4) 交互性 用户能进行人机对话,交互方式工作,联机地控制程序。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989 (郑宇华)

fenshi gongxiang moshi

分时共享模式 (time-sharing mode) 在计算资源集中的主机上,所有的程序都在主机上运行的环境下,多个用户同时通过终端共享主机计算资源的一种计算模式。这是 20 世纪 70 年代个人计算机出现之前流行的一种计算模式。终端的功能仅仅是一个用户与主机的接口,没有计算功能。主机上的分时操作系统将主机的计算资源按照预定的策略分配给用户。

分时共享模式中的主机一般被用来运行巨大的作业和保存大量关键数据。主要用于银行业、公共服务业和信息服务业。

新型的主机系统多采用并行系统。虽然从 20 世纪 90 年代开始,分时共享模式被客户-服务器模式逐渐取代,但是近几年,由于新的主机软件体系结构的出现,这种计算模式又重新得到应用(参见客户-服务器计算)。

分时共享模式的优点是计算资源集中,便于维护。它的缺点是实现图形用户界面的难度很大,而且很难进行用户应用程序的维护。(王勇)

fenxi xuexi

分析学习 (analytic learning) 以演绎为主要手段,利用现有知识对问题进行分析求解来获取知识和经验的一种学习方法。20 世纪 80 年代开始,机器学习研究进入了一个蓬勃发展的时期,出现了各具特色的一些学习方法和系统。人们在研究中发现有些学习方法在某些方面具有共性,例如解释学习,事例学习,类比学习等。它们的共同特点是:依靠大量已有的知识,且以演绎手段为主。人们把这一类方法统称为分析学习。分析学习是与归纳学习、遗传学习、连接机制学习等并列的一类机器学习方法。

基于解释的学习 (EBL) 一种从单例出发利用领域知识进行分析学习的方法。EBL 系统解释为什么一个特定的例子是一个概念的实例,然后将这个解释概括成可操作的概念识别规则。从例子学习长期以来是机器学习领域的研究重点。EBL 即是一种

示例学习的方法,但它与早期的示例学习方法有很大的差别。早期的示例学习研究主要集中在归纳方法上,从大量例子中抽取概括它们共同的特征,这种归纳方法是基于经验的、数据密集型的方法,缺乏知识的指导。80 年代以来,许多研究工作集中到分析的方法上来,例子不再被看作是孤立的特征集合,而是放在领域知识背景下进行处理。EBL 能够通过分析从单例中抽象出概念的描述,而这种分析过程是由领域知识引导的。例子的解释过程确定了例子中与概念相关的特征,这些相关特征构成了概念的充分条件。由 EBL 概括而生成的概念描述使得对以后的例子的识别更方便、有效。因此 EBL 从本质上说是基于演绎的、知识密集型的学习方法。EBL 代表了人工智能研究向基于知识的系统转变的一般趋势。EBL 的发展演变是许多学者共同努力的结果,最早可以追溯到早期的分析学习程序,如 STRIPS, HACKER 等,这些程序通过以往的经验来改善系统的执行。但这些工作并没有系统地探讨这种分析式的学习方法。直到 80 年代初,一些学者如 Dejong 等分别独立地在各自不同的应用领域中对这种学习方法进行了研究,强调了基于知识的单例学习。最后,一系列的比较、综述性的文章试图将这些工作统一到一种单一的学习方法之中。1986 年 Dejong 提出了 EBL 这个术语用来命名这种学习方法。在以后的一段时间里 EBL 成为机器学习中的研究热点。Mitchell 阐述了 EBL 的一种学习算法——EBG 学习算法,该算法内容如下。

已知:

目标概念 以高级形式或功能特性描述的某个概念的定义,它不能直接用于目标概念的实例的识别。

训练例子 以低级特性描述的目标概念的一个实例。

领域知识 是推理规则和事实集,用于证明训练例子是否是目标概念的实例。

可操作性准则 为使目标概念能被有效识别而规定的概念定义的表达形式。

要求: 一个训练例子的抽象,必须是目标概念的充分的概念定义,且满足可操作性准则。

算法: 第一步,利用领域知识构造解释,证明训练例子是目标概念的一个实例;第二步,对这个解释作概括,以获得一个一般的充分条件集以便用于同类例子的识别,这一过程采用目标回归的方法。Dejong 提出了一种更有效的 EGGS 方法,并将两步合

为一步。

EBL通过对单例的解释和概括,将不可操作的概念描述转化为可操作的概念描述,虽然没有超越原有的知识闭包,但通过对原有的知识进行重组,使得系统在以后完成相同或相似的任务时运行效率得到提高。训练例子被用来引导对可操作定义的搜索,从而使搜索更容易控制;同时例子作为以后将要识别的概念的代表,使得学习到的目标概念的描述,将来是有用的。EBL研究的问题包括:领域知识不完善问题,可操作性问题,效用问题,与其他方法的结合以及目标概念的问题,如确定目标概念和从失败中学习。

事例学习 是一种使用事例推理的分析学习方法。事例推理方法对训练事例集合进行组织记忆和索引;对于待求解的新事例,从事例记忆库中寻找与其相似的事例,利用它提供的有关信息对新事例的求解加以指导。事例推理思想源于 R. C. Schank 于 1982 年所著的《动态记忆》一书。1986 年,他在《解释模式》中建立了更加完善的事例记忆、检索、解释和推理模型。1983 年, Cyrus 系统在计算机上首次实现了《动态记忆》中的许多原理。其后在法律、烹调、医疗、管理等领域又研制出成功的 CBR 系统。20 世纪 80 年代末开始兴起事例推理研究与应用的热潮。根据输入问题的特征,事例推理系统的索引机制在事例库中搜索相似的候选事例集合,由其与问题的匹配程度决定可用的事例,然后运用相似事例的有关信息指导问题的求解。问题得到解答后,若系统认为具有保存价值,则将其存入事例库中,并为其建立索引信息。随着求解和学习活动的不断进行,保存的事例越来越多,求解能力也不断增强。当

前事例推理研究的主要内容有:①事例表示。寻找适合作为事例的语义和存储单位,常用的有实体—属性—值、框架、脚本、对象、记录等;②事例存储、索引与检索。寻求根据事例间语义相似性进行非精确的、索引与搜索相结合的查询能力,常用的技术有最小相邻法、归纳法和基于知识的索引法等;③事例、索引的增加与调整。探求在事例增加过程中如何补充不完善事例以及改善检索的针对性、灵活性和效率。

类比学习 也是一种分析学习方法(参见类比学习)。

参考文献

石纯一、黄昌宁、王家祯. 人工智能原理. 北京:清华大学出版社, 1993

fenzu jiaohuan

分组交换(packet switching) 存储转发交换技术中的一种。它是将要传送的报文分割成许多具有统一格式的报文分组,简称分组,并以此为传输的基本单元进行存储转发。

按对分组传输路径的管理不同,分组交换又可分为虚电路和数据报两种方式。**虚电路**是通信双方在进行通信时首先在它们之间建立一条逻辑电路,在一次通信中,该电路为通信双方所独占,这一点和**电路交换**相同。但从物理上看,该逻辑电路所经过的物理信道同时可为其他通信用户所共享。也就是说,物理信道可被多对通信用户复用,物理信道上交织传输着不同用户对的分组信息,如图 1 所示。

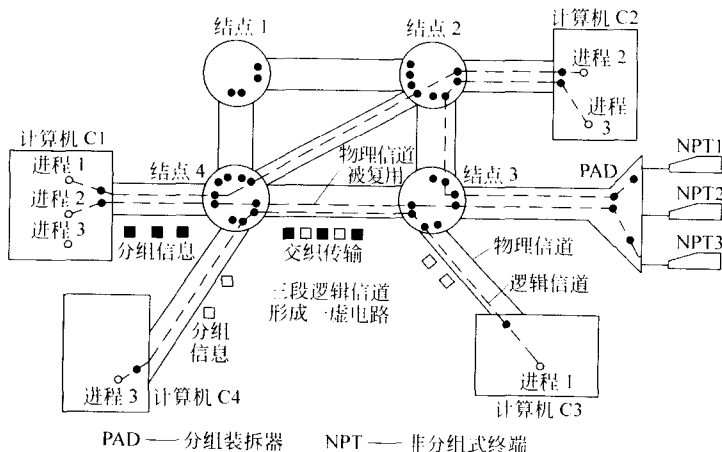


图 1 分组交换工作原理

虚电路是在每次通信时通过通信双方呼叫和协商建立,然后传送数据,通信完毕即行拆除,这个过程称为一次虚呼叫。这种虚电路称为**呼叫虚电路**(CVC),也称为**交换虚电路**(SVC)。如果通信双方的虚电路是被永久分配定的,则这种虚电路称为**永久虚电路**(PVC)。无论是 SVC 还是 PVC,它们都是由多段逻辑信道组成。物理信道可以分为多个逻辑信道,构成多个虚电路,从而实现物理信道的复用。在使用虚电路的情况下,一次通信中的各分组都是沿着该条虚电路传输的。而数据报的每个分组传输路径不是沿着固定的一条虚电路,而是每经过一个交换机就要动态地选择一次,这同报文交换的存储

转发交换方式路径选择相似,故称为**数据报**。

分组交换采用的协议是 X.25 建议。

X.25 建议

X.25 建议是国际电报电话咨询委员会(CCITT)所制定的用于分组交换网中连接**数据终端设备(DTE)**和**数据电路设备(DCE)**之间的接口规程标准。该标准 1974 年开始制订,1976 年获得通过成为一项国际标准。之后,每隔 4 年对它作补充修订。比较通用的是 1984 年和 1988 年版本。

X.25 建议定义了 OSI 网络体系结构的下 3 层协议,即 DTE 与 DCE 之间的物理层、数据链路层和网络层的协议,如图 2 所示。

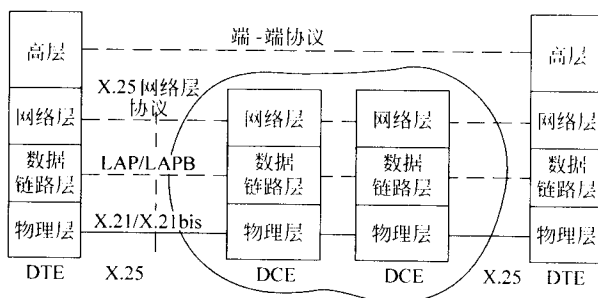


图 2 X.25 协议体系结构

物理层主要定义 DTE 和 DCE 之间接口的机械的、电气的、功能的和规程的特性。

X.25 物理层所采用的标准有 X.21 建议、X.21bis 建议和 V.24/RS232C 等 3 种,后两种实际上是兼容的(参见**数据通信接口标准**)。

数据链路层协议也叫帧级协议,因为这一层的信息传送以帧为基本单位。它是利用物理层提供的比特流传输功能,在 DTE 和 DCE 之间进行透明的、可靠的数据(帧级)传输。X.25 数据链路层的标准包含高级数据链路控制(HDLC)规程的子集 LAP(链路访问规程)和 LAP B(平衡型链路访问规程)。LAP B 受到了特别推荐,其主要功能是:

(1) 可以通过帧格式中的帧发送序列号和接收序列号使帧接收与帧发送顺序相同,以保持帧的原有顺序;

(2) 采用窗口机制进行数据流量控制和帧的接收与发送同步;

(3) 采用帧检验序列进行差错检测和纠正,保证可靠的数据传输。

网络层提供虚电路服务(虚电路建立、维护和释放)、寻址与**路由选择**、流量控制、顺序检验和错

控制等,以保证网络用户(DTE 与 DTE)间透明可靠的数据传输。网络层的主要功能为:

(1) 在 X.25 接口处为每个用户呼叫提供一个逻辑信道;

(2) 为每个用户的呼叫连接提供有效的分组传输,包括分组顺序编号,分组的确认和流量控制等;

(3) 提供呼叫虚电路和永久虚电路的连接。对于呼叫虚电路,要提供建立和清除呼叫虚电路连接的方法;

(4) 检测和恢复分组的差错。

分组交换网交换技术的新发展

分组交换网是分组交换结点的分布集合,如图 3 所示。在分组交换中,每个分组包含一部分用户数据加上一些控制信息。控制信息至少应包括分组被传送到它的目的地所需信息。在通过的每个结点,分组被接收、存储,再发送到下一个结点。例如,站 A 要发送一个分组给站 E。分组中应包含有目的地 E 的控制信息。把分组先从 A 点发往结点 4。结点 4 存储该分组,并且决定下一步送到结点 5,于是把分组放到发往结点 5(即 4—5 链路)的队列中。

当链路可用时,把分组传输到结点 5,然后再经过结点 6,最后到达 E。

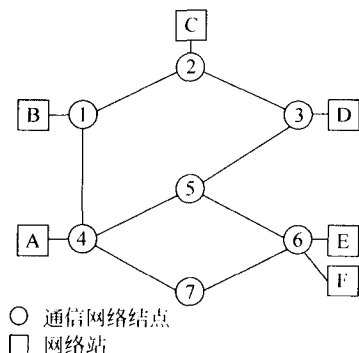


图 3 分组交换网络

X.25 建议这一传统的分组交换技术是在通信网以模拟信道为主的年代制定的。这种信道的数据传输速率一般不大于 9.6 kb/s, 误码率较高, 一般为 $10^{-5} \sim 10^{-4}$, 不能满足数据通信要求。X.25 建议一方面实现信道的多路复用, 同时进行差错检测和纠正, 使误码率降到小于 10^{-10} 的水平, 从而满足了绝大多数数据通信的要求。但是, X.25 建议是个比较复杂的建议, 以此为基础的分组交换网提供的是一种中、低速数据通信业务。随着分组交换技术的发展, 分组交换网性能不断提高, 分组交换机之间的中继线传输速率由 9.6 kb/s 提高到 2.048 Mb/s。到了 20 世纪 90 年代, 用户对数据通信网的传输速率提出了更高的要求, 而已有分组交换技术的分组交换网的能力已几乎达到了极限。同时, 由于光纤通信技术的发展, 通信线路传输速率大大提高, 误码率大大降低, 因此导致了快速分组交换技术的出现。快速分组交换的特点是简化通信协议和发展高速交换设备。广泛采用的技术途径有两种: 一种称为帧中继技术(参见帧中继交换), 另一种称为异步传送模式(ATM)。

参考文献

1. 杜治龙. 分组交换工程. 北京: 人民邮电出版社, 1993
2. 高星忠, 陈锦章, 张有材. 分组交换. 北京: 人民邮电出版社, 1993
3. Behrouz Forouzan etc. 数据通信与网络. 潘乞, 朱丹宇, 周正康译. 北京: 机械工业出版社, 2000 (史美林)

fenzu zhuangchaiqi

分组装拆器 (packet assembler/disassembler, PAD)

用以支持非 X.25 数据终端设备(简称非 X.25 DTE)接入分组交换网的一种设备。它置于非 X.25 DTE 与分组交换网之间, 执行规程转换器的任务, 也就是把各种不同规程的 DTE 都统一转换成 X.25 规程, 以使它们之间实现互通。图 1 给出了 PAD 应用的各种情况, 其中非 X.25 DTE 是分成非 X.25 主机和非分组式终端两种情况来描述的。

多数的非分组式终端是字符终端, 它们能发送和接收起止式字符, 采用异步通信方式, 因而也常被称为起止式终端或异步终端。对于使这类终端接入分组交换网的 PAD, 国际电报电话咨询委员会 (CCITT) 特别制定了 X.3, X.28 和 X.29 三个建议来规定其工作。这三个建议的主要功能为: ①提供对 X.25 规程支持, 用于与非分组式终端连接; ②向非分组式终端提供通过分组交换网建立呼叫、进行数据传输和清除呼叫的能力; ③向非分组式终端提供观察和修改接口参数的能力, 以适应不同终端的要求。

下面简单介绍这三个建议。

X.3 建议 此建议定义了一组参数(共 22 个), 这组参数规定了使非分组式终端接入分组交换网的 PAD 所执行的功能, 主要包括异步接口控制、数据转送控制、发送输出控制、发送输入控制、编辑功能

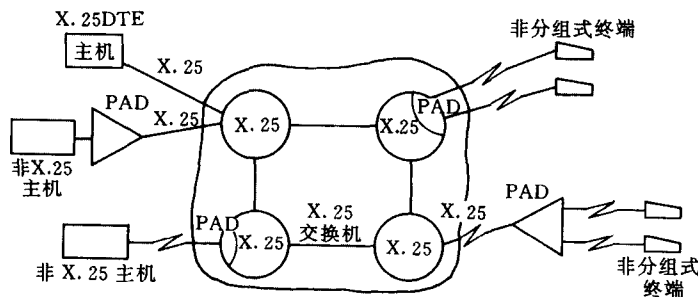


图 1 PAD 的应用情况

等。这些参数的值可以由网络操作人员设置,也可由用户使用 X.28 的命令进行修改,也可由远端的 PAD 或远端的分组式终端使用 X.29 建议的报文进行修改。

X.28 建议 此建议规定了起止式终端与 PAD 的接口规程。它主要包括以下 4 方面内容:

(1) 起止式终端与 PAD 之间的物理通路建立过程。规定了经由公用交换电话网或租用线路时需调制解调器的类型及其电气特性,或者经由公用数据网或具有 X 系列接口的租用线路的接口物理特性,还规定了通路的建立与拆除过程。

(2) 起止式终端与 PAD 进行通信的初始化工作。规定了用于控制信息交换的字符格式, X.3 建议的 22 个参数选择(即 PAD 轮廓值),通过业务请求信号使 PAD 知道终端所使用的数据速率、编码及

奇偶检验等。

(3) 起止式终端与 PAD 之间控制信息的交换过程。包括 PAD 命令信号、服务信号、呼叫建立和清除过程。

(4) 起止式终端与 PAD 之间用户数据的交换过程。

X.29 建议 此建议定义了分组交换网中一个 PAD 与远端 PAD 或 X.25 DTE 之间交换控制信息和用户数据的过程。其主要功能是在起止式终端与远端 PAD 或 X.25DTE 通信之前,使本地 PAD 与远端 PAD 或 DTE 之间交换信息,从而使远端 DTE 了解本地 DTE 的各个方面信息,并进行协调控制,使双方顺利完成通信。

图 2 给出了 X.3, X.28, X.29 和 X.25 等建议在分组交换网中的关系。

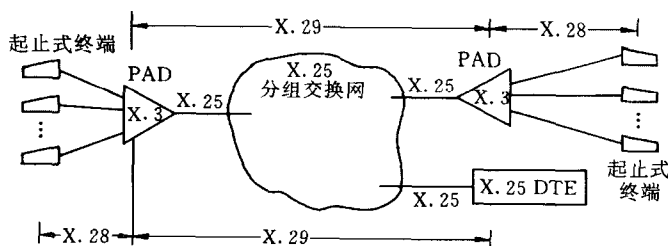


图 2 X.3, X.28, X.29 和 X.25 等建议在分组交换网中的关系

参考文献

1. 杜治龙. 分组交换网工程. 北京: 人民邮电出版社, 1993
2. 高星忠, 陈锦章, 张有材. 分组交换. 北京: 人民邮电出版社, 1993
3. 汪润生, 周师熊. 数据通信工程. 北京: 人民邮电出版社, 1990

(史美林)

转换成一种所封装的程序可用的形式);四是对特别难用的资源提供抽象,例如,程序人员可以使用的计算机的有些控制功能是通过操作系统接口提供的,这样,既安全又便于使用。

参考文献

- Ralston A et al. Encyclopedia of Computer Science. Third Edition. 1993

(徐家福)

fengzhuang

封装 (encapsulation) 一种用以隔离书写程序中所作的某些决断的技术。这些决断有数据表示,算法实现等等。为此,往往将程序组织成接口和内部两部分,只是接口对用户可见,内部对用户是隐蔽的。

封装的作用有四:一是提供一种信息隐蔽技术;二是用以强制推行特定的访问风纪,例如,通过管程访问程序的临界区;三是在程序间提供某种相容性,后者是通过一个接口来实现的(接口将控制和数据

fuwuqi

服务器 (server) 在网络环境中或在具有客户-服务器结构(参见客户-服务器计算)的分布式处理环境中,为客户的请求提供服务的结点计算机。客户-服务器是实现资源共享的一种结构,客户是服务器的服务对象。在某种应用环境中的服务器,也可能在另外一种应用环境中成为客户。

服务器可以是微型计算机、工作站、小型计算机、大型计算机乃至大规模并行处理的高性能计算机。服务器软件主要包含操作系统、网络协议、数据

库管理系统以及各种开发工具与软件中间件。这些软件用来支持客户和服务端之间相互作用,负责透明地连接客户和服务端系统。服务器可提供文件、数据库、打印、通信、图形、图像、安全、保密、系统管理、网络管理以及信息发布等服务。按服务器的规模和性能可分为群组级服务器、部门级服务器和企业级服务器等。按服务器用途可分为专用服务器和通用服务器。此外尚有按客户与服务端之间的连接是否通过通信服务器而分为远程服务器和近程服务器。

服务器的主要特点有:①服务器只是在客户的请求下才为其提供服务,而不主动为客户提供服务;②透明性,即服务器对客户完全透明,一个与服务器通信的客户完全不必知道服务器的存在及其工作情况;③高性能、高速度、大容量、高可靠性及可伸缩性。

服务器的概念最早出现于局域网条件下的客户端-服务器结构。这时的服务器按功能可分为文件服务器、打印服务器、数据库服务器、应用服务器和通信服务器等。应用服务器又可分为计算服务器、决策支持服务器、联机事务处理服务器等。

文件服务器是服务器中最早出现的一类,它主要为用户提供文件存储和共享服务,文件服务器一般具有较强的存储能力和较高的通信速度,借此为客户提供高速和大容量的文件存储服务。打印服务器则是用于打印服务的专用服务器。通过使用打印服务器的服务,客户端可以共享高速打印资源。数据库服务器是服务器中专门提供数据库服务的一类服务器,它为网上客户提供数据库的查询、更新、索引、事务管理等服务。通信服务器则是为客户端提供各种通信转发、中继等方面服务的服务器。

随着因特网的发展和普及,作为因特网一个重要组成部分的因特网服务器也日益受到重视。通过因特网向分布在全球的用户提供各类信息服务的服务器叫做因特网服务器。因特网服务器可以提供的服务包括将按树状结构分层的域名解析为因特网地址的域名地址转换服务(DNS)、完成电子邮件的传输与管理的电子邮件服务(E-mail)、文件传输服务(FTP)、电子公告板服务(BBS)、广域信息服务(WAIS)、网络新闻服务(USENET)、万维网服务(Web,WWW)等。一台因特网服务器可以同时提供一种或多种上述服务。

服务器的发展大致经历了3个阶段,第一阶段

为20世纪80年代前期,产生了网络操作系统 Netware 和文件服务;第二阶段为20世纪80年代后期,产生了客户-服务器结构和数据库服务器;第三阶段为20世纪90年代至今,是服务器大发展的时期,并向分布、面向对象、智能化和开放的方向发展。同时,随着因特网的发展,因特网服务器和基于万维网服务技术的各种应用服务器也成为服务器发展的重点。

(黄震春)

fudianshu biao zhun

浮点数标准 (floating-point number standard) 由标准化组织制定的浮点数的表示格式和运算规则。

浮点数是实数的一种近似表示,浮点数格式必须兼顾表示范围及表示精度的要求。浮点数的运算规则必须尽可能地保持精度,缩小误差。在计算机诞生后的很长一段时间里,由于没有统一的浮点标准,不同系列的计算机采用各不相同的浮点表示形式,给数值计算和软件移植带来了困难。考虑到微处理器性能的不断提高和计算机应用的进一步普及,IEEE(电气和电子工程师协会)在20世纪80年代制定浮点标准,成为所有微处理器遵循的二进制浮点算术运算标准,即IEEE 754标准(IEEE的另一个854标准主要针对十进制浮点运算,几乎很少被使用)。

IEEE 754标准于1985年3月获IEEE标准委员会批准,同年7月成为ANSI(美国国家标准学会)标准。标准的内容包括浮点数的表示形式、浮点操作的类型和定义、舍入方式、例外处理方法等。这里主要介绍浮点数的表示形式。

IEEE 754主要定义了单精度(32位)和双精度(64位)两种基本格式,以及扩充单精度和扩充双精度两种扩充格式,但对扩充精度仅指定了对精度的最低要求。计算机系统可以用硬件、软件或硬、软件结合的方式实现IEEE 754标准或标准的主要部分,且任何实现都必须至少包括单精度浮点格式。

在IEEE 754的浮点格式中,尾数用原码表示,指数用增码表示,各种格式的有关参数见表1。

基本格式由1位符号 s 、指数部分 e 和小数部分 f 组成,如图1所示。对单精度数, e, f 分别是8位和23位;对双精度数, e, f 分别是11位和52位。左边是最高位。此格式表示的数 X 的值 v 由如下规则确定。

表1 IEEE 754 标准定义的浮点格式参数

参 数	单精度	扩充单精度	双精度	扩充双精度
表示精度 / 位	24	≥ 32	53	≥ 64
最大指数	+127	$\geq +1\ 023$	+1 023	$\geq +16\ 383$
最小指数	-126	$\leq -1\ 022$	-1 022	$\leq -16\ 382$
指数偏置	+127	未指定	+1 023	未指定
指数部分位数	8	≥ 11	11	≥ 15
格式总位数	32	≥ 43	64	≥ 79

s	e	f
---	---	---

图1 IEEE 754 标准定义的浮点格式

对单精度数:

(1) 若 $e = 255$ 且 $f \neq 0$, 则无论 s 是什么, v 称为 NaN, 即非数 (Not a Number);

(2) 若 $e = 255$ 且 $f = 0$, 则 $v = (-1)^s \times \infty$, 即正无穷或负无穷;

(3) 若 $0 < e < 255$, 则 $v = (-1)^s \times 2^{e-127} \times (1.f)$;

(4) 若 $e = 0$ 且 $f \neq 0$, 则 $v = (-1)^s \times 2^{e-126} \times (0.f)$;

(5) 若 $e = 0$ 且 $f = 0$, 则 $v = (-1)^s \times 0$, 即零。

对双精度数: 将上面的 255, 127, 126 分别替换成 2 047, 1 023 和 1 022 即可。

上述定义表明, 除零有正零和负零两种形式外, 其他数的表示都是惟一的。大部分浮点数都采取由 (3) 定义的规格化数的形式, 其尾数的最高位为一隐含的 1, 从而 23 位小数部分可表示 24 位精度。作为一种特例, (4) 既解决了 (3) 不能表示零的问题, 又能充分利用有限的空间表示更多的数。由 (4) 表示的数称为反规格化 (denormalized) 数。在实际运算中, 无穷和 NaN 都属于非法操作数, 其中无穷可能来自被零除或上溢, NaN 可能是 $\sqrt{-1}$ 等操作的结果。当出现非法结果或非法操作数时, 标准规定系统可以自行决定是产生中断还是继续运算。由于在 (1) 中定义的 NaN 只要求 f 为非零值, 在允许产生中断的实现中, f 可用于存放系统状态或中断原因, 供中断处理程序使用。

IEEE 754 标准规定的舍入方式共 4 种, 即向最

近数、向 $+\infty$ 、向 $-\infty$ 以及向 0 舍入, 其中向最近数舍入是默认的舍入方式。在默认情况下, 若结果与最近的两个可表示数的距离相等, 则取最低位为零者。

参考文献

1. IEEE. IEEE Standard for Binary Floating-Point Arithmetic. ACM SIGPLAN Notices, 1987, 22 (2): 9~25
2. David Goldberg. What Every Computer Scientist Should Know about Floating-Point Arithmetic. ACM Computing Surveys, 1991, 23(1): 5~48 (唐志敏)

fushedu jishu

辐射度技术 (radiosity technique) 基于热能辐射原理, 在封闭环境中求取物体表面光能分布的一种全局光照技术。该技术于 1984 年提出。它特别适用于由漫反射面组成的封闭环境。辐射度技术是根据光能能量传播平衡的原理计算环境中每一平面片上的光照能量。利用该技术可以很自然地形成物体之间的颜色渗透现象。

辐射度技术所考虑的是具有各向相同的反射特性的漫反射物体, 即表面不甚光滑的物体。这些物体对于光照在各个反射方向上具有等量均匀的反射光亮度, 因而辐射度计算的辐射光亮度与观察者的位置即视点无关。经典的辐射度技术是将环境中的平面 (曲面) 划分成足够小的面片, 而后根据环境中光能传播平衡的原理建立线性方程组求解每个面片上的光强。

光能传播和辐射计算过程中所涉及的一个重要问题是, 对于一个特定的具有光能的面片, 它所传播出的光能中究竟有多少比值的能量能够到达另一特定面片。在辐射度技术中这一比值称为形状因子。形状因子的大小只与环境中的物体的几何状况 (大小、位置) 有关, 因而一旦造型环境确定, 形状因子即可预先计算。通常在辐射度求解中其形状因子的计算占用绝大部分开销, 因而形状因子的计算效率对于辐射度计算举足轻重。在 1985 年提出了一种效率较高的形状因子计算技术, 称为半立方体方法。在利用该方法求取从一个面片 (如面片 i) 到其他面片的形状因子时, 在此面片中心放置一个单位半立方体, 然后将其他面片向半立方体各面上进行投影, 以求取形状因子的大小。

作为一种成功有效的全局光照技术, 辐射度技术自 1984 年以来在多方面取得重大进展。首先, 为了使辐射度技术解决复杂环境中的应用问题, 1988

年出现了逐步求精的辐射度技术;为了使辐射度技术能够处理包括非漫反射物体的环境,提出了多种非漫反射环境的辐射度技术;在辐射度求解的过程中,环境物体需划分成作为求解单元的小面片,为了提高效率和改善图形质量,提出了有效地组织和划分面片的层次辐射度技术。常规的辐射度技术是针对平面(曲面)环境设计的,而将该技术推广到更复杂的环境尤其重要。当前辐射度技术已经成功地推广到具有参与介质(云雾、灰尘等)、几何纹理面(橘子皮等)、分数维几何面(山脉等)、毛绒表面等复杂环境。

参考文献

1. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2001
2. Cohen M F, Wallace J R, Hanrahan P, Greenberg D P. *Radiosity and Realistic Image Synthesis*. New York:Academic Press Professional, 1993

(吴恩华)

fuzhu cunchuqi

辅助存储器 (auxiliary memory) 在计算机存储系统中不直接向中央处理器提供指令和数据的存储设备。主存储器存取速度快,但容量相对较小。辅助存储器的存储容量大,存储1个数据所需的费用低,在存储系统中起扩大容量的作用。

一个计算机系统的辅助存储器由一种或多种存储器组成,如图1所示。在辅助存储器中,磁盘存储器的存取时间短,存储容量大,是辅助存储器的主要存储设备。磁带存储器的联机存储容量比磁盘存储器的小,存取时间也长。但是磁带易于脱机存放,而且可以和其他计算机的磁带互换使用,往往用于存放长期保存的数据,充作档案库存储器。光存储器的存储容量大,存取速度接近磁盘存储器,但它不易写入,多采用只读的工作方式,用于存放大量程序和数据。磁盘、磁带和光盘存储器在向中央处理器提供数据时,先把数据送到主存储器,再由中央处理器调用。海量存储设备存储容量极大,但存取时间最长。它按操作系统调度的要求,自动地把中央处理器需要的数据传送到磁盘存储器,供主存储器调用。它用于巨型数据库和大型计算站等要求特大联机存储容量的场合。

辅助存储器是计算机外围设备的一部分,中央处理器通过通道和中断系统控制它的工作。

数据在辅助存储器中是以记录块为单位进行存取的。在存取时,按中央处理器给出的块地址找到

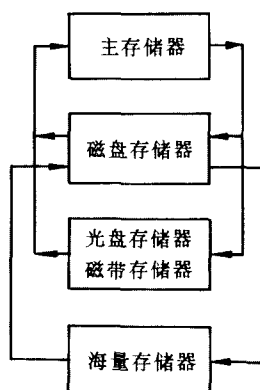


图1 由磁盘、磁带和海量存储器组成的辅助存储器

所需的记录块,用成块方式与主存储器交换数据。按照寻找记录块的方法,辅助存储器的存储设备分为两类:①顺序存取存储器。存储器的读写机构从所在的记录块开始,顺序查找,直到找到所需要的块,磁带存储器属于这一类。②直接存取存储器。存储器的读写机构按记录块的地址直接寻找到一个较小的区域,然后在这个区域内顺序寻找所需的记录块。例如,在磁盘和光盘存储器中先找到柱面,再在柱面内找到所要的块。直接存取存储器的存取速度比顺序存取存储器的快。

应用于辅助存储器的存储设备大多是磁表面存储器,它以涂覆于非磁性材料表面的磁性薄层作为存储媒体。20世纪80年代中期以来,光盘存储器得到迅速发展。

(郑衍衡)

fuwang cunchu

附网存储 (network attached storage, NAS)

一种将存储设备直接连网并使用标准协议提供文件级数据访问的存储方式。在传统方式中,数据请求经过服务器发给存储设备,存储设备通过服务器上网。采用附网存储方式的存储设备自身具有网络接口,能直接连接到诸如以太网的各种网络。标准协议是指网络文件系统(NFS)、通用互连文件系统(CIFS)等。文件级数据访问指数据请求是以文件句柄及偏移值为参数的访问方式。附网存储的突出特点就是在物理连接上将存储器直接连接到网络上,不再挂在服务器后端,避免了给服务器增加I/O负载。

采用附网存储方式组成的存储系统称为附网存

储系统,采用附网存储方式的存储设备称为附网存储设备。

1992年,美国加州大学的 Randy H. Katz 教授首次提出了附网存储。在以后的几年中,附网存储概念逐渐被人们所接受。由于附网存储设备的易扩展性与易管理性,逐步形成了一个颇具规模的附网存储产品与服务市场。

随着数据量的不断增大,计算机技术正在经历以计算、传输为重点向以数据存储为中心的转变。以总线连接存储设备的方式在可扩展性方面受到很多制约,因此人们考虑在网络上建立专门提供数据存储功能的数据中心以满足扩充存储容量和提高存取速度的不断需求。虽然服务器模式能够满足各种网络服务的需要,但由于服务器功能面向通用计算,对于只需数据存储服务来说,软硬件资源浪费很大。附网存储将存储设备与服务器分离,采用瘦服务器形式,精简传统服务器各部分配置,只实现文件共享服务,因此成本相对较低。它以数据为中心,集中管理数据,从而有效利用带宽,提高网络整体性能。

附网存储系统作为网上的专用文件服务器,在网络环境下提供文件服务。它能在 UNIX 和 Windows NT 等多种网络环境下工作。为实现系统兼容,附网存储系统软件具备多功能层次,如图 1 所示。存储设备驱动层完成各类存储设备的底层驱动,并向上提供接口,以便于卷管理器统一对存储设备进行组织和管理。附网存储系统的可安装文件系统提供与其自身文件系统同种类型的服务,并且通过集成其他访问方法来扩展自身文件系统功能。应用层可以是 FTP 服务、Web/E-mail 服务或媒体流服务等。TCP/IP 驱动程序则使附网存储系统满足一般网络传输协议的要求。

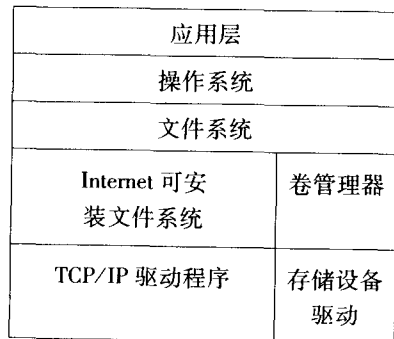


图 1 附网存储系统软件层次图

附网存储设备有如下特点。

(1) 可扩展性强

将附网存储设备连接到网络上非常方便。附网存储设备提供 RJ-45 接口和单独的 IP 地址,可以将其直接挂接在主干网的交换机或其他局域网的集线器上,通过简单的设置(例如设置 IP 地址等)就可以在网络上使用,且即插即用,在线扩容无需停顿网络。因此,与传统的服务器和直接存储设备相比,附网存储设备具有容量扩展能力强的优势。

(2) 使用和管理简单

附网存储设备的安装、调试、使用、维护和管理非常简单。一台附网存储设备占用一个 IP 地址,实质上相当于一台高性能的文件服务器,却可以大幅节省设备费用。另外,附网存储设备能完全融合已建立起来的网络设备和协议,它作为独立的数据存储设备与其他各种服务器搭配,既保护了用户的原有投资,又将整个网络的性能提高到一个新的层次。

(3) 跨平台文件共享

以 UNIX 中的 Solaris, HP-UX, 以及 LINUX, FREE BSD 等为操作系统的附网存储设备通过支持网络文件系统协议实现对附网存储设备的文件共享。Windows 平台下的网络文件共享则采用通用互连文件系统,从而能够实现不同网络环境下用户跨平台共享数据。

(4) 性能优化

附网存储设备内置有优化的独立存储操作系统,可以有效地调度系统总线资源,全力支持 I/O 存储,因此它的效率较直接存储设备高,一般可高出 60%。同时,它面向存储管理,可以不经服务器将其中的重要数据进行本地备份。

附网存储设备主要通过提供冗余容错的磁盘阵列来满足 7×24 小时的长时间、高可靠的应用,并能通过 E-mail 系统将报警信息自动发给系统管理员。同时,它还能进行动态监测,并提供详细的安全日志报告,以求全面地保护珍贵的数据。

附网存储主要有附网存储硬盘服务器和附网存储光盘镜像服务器两种。前者提供可扩展的在线存储能力,后者提供离线存储服务。附网存储硬盘服务器是一种将廉价冗余磁盘阵列(RAID)技术与文件服务相结合的技术。附网存储光盘镜像服务器则是将硬盘高速缓存和瘦服务器技术相结合、专为光盘网络共享而设计的网络共享存储设备。它使客户机能以硬盘的访问速度来共享光盘上的信息资源,

以消除 CD-ROM 驱动器速度瓶颈问题,改善了光盘网络共享的性能,适合于图书馆等资料众多且用光盘保存文件的应用场合。

附网存储已被广泛应用到如下许多领域:互联网服务提供商(ISP),应用服务提供商(ASP),计算机辅助设计与计算机辅助制造(CAD/CAM),大中小型企业,大中小学和图书馆、出版社,多媒体与影视动画的制作单位,广告公司,政府、军队、银行等行业用户以及航空、医疗等行业用户等所有需要快速大容量存储设备解决方案的场合。

附网存储迎合了网络技术的飞速发展和以数据为中心的技术趋势,在计算机存储技术领域占据一席之地。

参考文献

1. Randy H Katz. Network-Attached Storage Systems. Proceedings of the Scalable High Performance Computing Conference-SHPCC-92 1992, Published by IEEE. 1992: 68 ~ 75

2. 张江陵,冯丹.海量信息存储.北京:科学出版社,2003 (周可)

fuzaxing duliang

复杂性度量 (complexity measure) 算法复杂性的定量描述。算法的复杂性可由不同的标准来衡量,例如描述算法所用语言的长度,称为描述复杂性。解决一个问题的各种算法程序的长度的下界称为该问题的**复杂性**(又称 Kolmogorov 复杂性)。但最重要的复杂性度量是执行算法所耗用的资源量。一般说来,处理规模较大的输入比规模较小的输入要耗用更多的资源,这里“资源”一词主要意指时间和存储空间。用算法耗用资源依赖输入规模的函数来表示算法的复杂程度:

$f(x, C)$ = 输入为 x 时,算法 A 所耗费的资源量
通常称 $f(x, C)$ 为算法 C 的对输入 x 的复杂性量度。
记输入规模为 x 的长度 $|x| = n$, 称

$$W(n, C) = \max \{ f(x, n) \mid |x| = n \}$$

为输入规模为 n 时,算法 C 的**最坏情况复杂性**。又若已知输入规模为 n 的各个输入 x 的概率分布 $p_n(x)$, 则称

$$A(n, C) = \sum_{|x|=n} p_n(x) * f(x, n)$$

为输入规模为 n 时,算法 C 的**平均情况复杂性**。

对于一个问题,它有各种各样的算法,我们称 $f(n) = \min \{ W(n, C) \}$ 或在一定模型下解决同一问

题的各个算法 C 为该问题的固有计算难度,或称该问题的计算复杂性。实际上,计算机理论的重要分支算法设计和分析是对各个具体问题寻找复杂度尽可能低的算法 C , 并求出 $W(n, C)$ (或 $A(n, C)$), 它是该问题的复杂度 $f(n)$ 的上界,从而得知该问题应属于的复杂性类(参见**多项式谱系**)。同时人们研究 $f(n)$ 的下界,从而得知该问题不可能属于的复杂性类,不再去设计复杂度比下界更低的算法。

从算法分析的实践中可以看出,人们感兴趣的是 $f(n)$, $W(n, C)$, $A(n, C)$ 在 n 趋于 ∞ 时的渐近性态,它们的常数项或系数是一个比较次要的因素,人们更为关心的是它们的增长率,简称阶。下列符号用于表示函数的阶(定义中的函数都是从正整数映射到正实数)。

(1) 集 $O(f)$ 中的任一函数 g , 存在一个常数 $r > 0$, 使得对于所有的 n , 有 $g(n) < rf(n)$ 。

(2) 集 $\Omega(f)$ 中的任一函数 g , 存在二个常数 $n_0, r > 0$, 使得对于所有充分大的 $n > n_0$, 有 $g(n) > rf(n)$ 。

(3) 集 $O(f)$ 中的任一函数 g , 有

$$\lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = 0$$

(4) 集 $\Theta(f)$ 中的任一函数 g , 有 $g \in O(g)$ 且 $g \in \Omega(f)$ 。

在复杂性中最常出现的阶函数有: $\log^k n, n^k, n^{\log n}, 2^n$ 等, 相应阶的函数依次被称为: 对数阶, 多项式阶, 亚指数阶, 指数阶。计算机科学家发现一个问题在一种计算模型下可以用多项式阶或指数阶的算法求解, 那么在别的计算模型下也可以用多项式阶或指数阶的算法求解。称之为**相似性**和**对偶性原理**。因此算法复杂性的阶是一个独立于计算模型的, 而问题的复杂性的阶又是独立于算法和计算模型的仅由问题本身所决定的重要特性。

近年来, 人们鉴于最坏情况复杂性研究难有进展, 转向平均情况复杂性的研究; 从确定型的算法复杂性研究转向不确定的、不精确型的概率算法(例如退火算法)复杂性研究; 从串行算法转向并行型算法(例如遗传算法)。这些新型算法复杂性度量定义也要作相应的调整。问题的复杂性度量有时得到一些实用上的改善, 但理论上, 只要计算模型没突破图灵机模型, 同一问题的计算复杂性度看来无望有质的飞跃。

参考文献

1. Li M, Vitanyi P. An Introduction to Kolmogor-

ov Complexity and Its Applications. Springer Verlag, 1993

2. Balcazar J L, Diaz J, Gabarró J. Structural Complexity, I, II. Springer Verlag, 1988

3. Hong Jia-Wei. Computation: Computability, Similarity and Duality. London: Pitman, 1986 (朱洪)

fuzaxing guiyue

复杂性归约 (complexity reduction) 计算复杂性理论中研究问题复杂性类之间关系的重要方法。复杂性归约是这一方法在计算复杂性理论中的应用和发展。S. A. Cook 于 1971 年首次使用这个方法证明了第一个 NP 完全问题。在计算复杂性理论中,用各种计算模型定义了多种复杂性归约,最常用的是图灵归约和多一归约。

图灵归约 设判定问题 Π_1 和 Π_2 , S 是关于 Π_2 的假想的“算法”。关于 Π_1 的以 S 为假想“子程序”的算法 A 称作 Π_1 到 Π_2 的图灵归约。若存在 Π_1 到 Π_2 的图灵归约,则称 Π_1 可图灵归约到 Π_2 , 记作 $\Pi_1 \leq_t \Pi_2$ 。如果进一步限制 A 具有某种计算复杂性 α , 则称 A 是 Π_1 到 Π_2 的 α 图灵归约。用 \leq_α 表示“可 α 图灵归约”。这里在计算 A 的复杂性时不考虑假想“子程序” S 的消耗,仅把调用一次 S 算作一步。当 α 为多项式时间界限时,称作多项式时间图灵归约,记作 \leq_p 。多项式时间图灵归约是 1971 年 S. A. Cook 提出的(参见图灵归约)。

复杂性归约可以用来比较问题的计算难度。取图灵归约的计算复杂性 α 足够的低。设 $\Pi_1 \leq_\alpha \Pi_2$, 那么如果 Π_2 有某种复杂性的算法,则 Π_1 也有这种复杂性的算法。或者反过来说,如果 Π_1 不存在某种复杂性的算法,则 Π_2 也不存在这种复杂性的算法。在这个意义下, Π_2 不比 Π_1 容易。例如,设 A 是一个多项式时间图灵归约。假设 S 是一个多项式时间算法,那么将 S 使用的时间计算在内, A 仍是一个多项式时间算法。因此,设 $\Pi_1 \leq_p \Pi_2$, 那么如果 Π_2 是多项式时间可解的,则 Π_1 也是多项式时间可解的;反之,如果 Π_1 不是多项式时间可解的,则 Π_2 也不是多项式时间可解的。从而,相对于多项式时间而言, Π_2 不比 Π_1 容易。设 \mathcal{C} 和 \mathcal{D} 是两个问题类。为了研究 \mathcal{C} 是否包含在 \mathcal{D} 中,只需考虑 \mathcal{C} 中“最难的”问题是否属于 \mathcal{D} 。复杂性归约为定义“最难的”问题提供了工具。如果 $\Pi \in \mathcal{C}$ 并且 \mathcal{C} 中所有的问题都可 α 图灵归约到 Π , 则称 Π 是 α 图灵归约

下 \mathcal{C} 完全的。 \mathcal{C} 完全的问题是 \mathcal{C} 中“最难的”问题,因为 \mathcal{C} 中的问题都不比它难。设 \mathcal{D} 在 α 图灵归约下是封闭的,即 $\Pi_1 \leq_\alpha \Pi_2$ 且 $\Pi_2 \in \mathcal{D}$ 蕴涵 $\Pi_1 \in \mathcal{D}$ 。那么,若 \mathcal{C} 完全的问题 $\Pi \in \mathcal{D}$, 则 $\mathcal{C} \subseteq \mathcal{D}$; 反之,若 $\mathcal{C} \not\subseteq \mathcal{D}$, 则 $\Pi \notin \mathcal{D}$ 。例如,多项式时间可解的判定问题类 P 在多项式时间图灵归约下是封闭的。设 Π 是多项式时间图灵归约下 \mathcal{C} 完全的,那么只要 Π 是多项式时间可解的,则 \mathcal{C} 中所有的问题都是多项式时间可解的;反之,如果已知 \mathcal{C} 中存在非多项式时间可解的问题,则 Π 是非多项式时间可解的。于是,复杂性归约为研究计算复杂性类之间的关系和问题的复杂性下界提供了新的途径:把问题类 \mathcal{C} 是否包含在 \mathcal{D} 中转化为是否存在 \mathcal{C} 完全的问题 Π 属于 \mathcal{D} ; 若已知 \mathcal{C} 不包含在 \mathcal{D} 中,只需证明 Π 是 \mathcal{C} 完全的就能推出 Π 不属于 \mathcal{D} 。

多一归约 一种常用的复杂性归约。设 Π_1 和 Π_2 是两个判定问题, f 把 Π_1 的每一个实例 I 变换成 Π_2 的实例 $f(I)$ 。如果对 Π_1 的每一个实例 I , I 的答案为“是”当且仅当 $f(I)$ 是 Π_2 的答案为“是”的实例,则称 f 是从 Π_1 到 Π_2 的多一归约。如果存在 Π_1 到 Π_2 的多一归约,则称 Π_1 可多一归约到 Π_2 , 记作 $\Pi_1 \leq_m \Pi_2$ 。和图灵归约类似,当限制 f 的计算复杂性为 α 时,称作 α 多一归约。当 f 为多项式时间可计算时,称作多项式时间多一归约,又称作多项式时间变换。它是 R. M. Karp 于 1972 年提出的。可以把多一归约看作图灵归约的特殊形式。设 f 是 Π_1 到 Π_2 的多一归约,如下构造 Π_1 到 Π_2 的图灵归约 A : 设 S 是关于 Π_2 的假想“算法”。对 Π_1 的实例 I , 先计算 $f(I)$, 然后对 $f(I)$ 使用 S 。 S 对 $f(I)$ 的回答即为 A 对 I 的回答。因此, $\Pi_1 \leq_m \Pi_2$ 蕴涵 $\Pi_1 \leq_t \Pi_2$ 。在计算复杂性理论,尤其在 NP 完全性理论中,广泛使用多项式时间变换。很多完全性都是在多项式时间变换下而言的。例如, NP 完全性在多数文献中是用多项式时间变换定义的。

对数空间归约 变换 f 的空间复杂性为 $O(\log n)$ 的多一归约,又称作对数空间变换。对于对数空间归约 f 还要添加一个条件:存在多项式 p 使得,对于 Π_1 的所有实例 I , $|f(I)| \leq p(|I|)$ 。这里 $|I|$ 和 $|f(I)|$ 分别表示 I 和 $f(I)$ 的大小。不可能利用多项式时间变换(或任何复杂性不低于多项式时间的归约)来进一步细分 P 类。例如要考虑 DLOG 与 P 的关系,其中 DLOG 表示所有空间复杂性为 $O(\log n)$ 的判定问题。已知 $DLOG \subseteq P$, 但不知道这个包含是否是真包含。为此,要定义 P 完全问题。

但是,这次不再能使用多项式时间变换,而必须使用对数空间变换,因为在多项式时间变换下 DLOG 不是封闭的(除非 $DLOG = P$)。 Π_1 可对数空间变换到 Π_2 ,蕴涵 Π_1 可多项式时间变换到 Π_2 。也有一些文献用对数空间变换定义关于 NP 等问题类的完全性。

此外,还有 γ -归约、强非确定型多项式时间图灵归约、随机归约、真值表归约等。复杂性归约不仅可以用于判定问题,也同样可以用于函数和搜索问题。

参考文献

1. Balcázar J L, Díaz J, Gabarró J. Structural Complexity I. Berlin: Springer-Verlag, 1988
2. Balcázar J L, Díaz J, Gabarró J. Structural Complexity II. Berlin: Springer-Verlag, 1990
3. Garey M R, Johnson D S 著. 计算机和难解性: NP 完全性理论导引. 张立昂, 沈泓, 毕源章译. 北京: 科学出版社, 1987 (张立昂)

fuza zhilingji jisuanji

复杂指令集计算机 (complex instruction set computer, CISC) 以微程序技术为基础具有较复杂指令系统的计算机。

复杂指令集计算机是相对于精简指令集计算机 (RISC) 而言的。在 20 世纪 60 年代到 80 年代初期这一阶段中,以微程序控制器(参见微程序控制器)为基础的处理机占主流地位。自 80 年代初、中期以后,采用硬连线控制器、且具有精简指令集的 RISC 处理机问世并迅速发展以后,就把过去二十多年中的传统的指令系统比较复杂的计算机称为 CISC,其处理机称为 CISC 处理机。

自 20 世纪 60 年代初 IBM 公司开始把计算机产品系列化并做到软件兼容以后,处理机体系结构设计中采用微程序技术作为控制指令执行的控制器的基础,是比较合适的。因为在一个产品系列中,低档机的指令系统中的指令的基本操作可以以微程序方式存放在微存储器中,这个微存储器称为“核”。如

果高档机要求增加功能更强的,更复杂的指令,则只需要扩充这个核,增加相应的微程序,即可做到从低档机到高档机的软件向上兼容。此外,以微程序技术为基础的控制器的实现是符合当时的计算机工艺的,因为在 70 年代末以前,计算机的主存储器仍为较慢的磁心存储器。当时,中央处理器 CPU 与微存储器都已采用双极型半导体集成电路或其他较快的电路,这种微存储器的周期与中央处理机的工作节拍相吻合,允许中央处理器 1 拍执行 1 条微指令。

指令系统逐渐变得复杂的原因很多,大致可归纳成 3 点: ①在产品系列中追求软件兼容性,如 VAX 的高档机要和 Micro VAX 兼容,Intel 80486 要和 Intel 8086 兼容。已有的即使不合理的指令仍要保留,而新的产品又要求增加一些新的指令。②认为指令系统愈复杂,就可以缓解所谓的软件危机,因此在指令系统中增加了接近于高级语言语句的指令,如 Call/Return 指令,这种指令的执行机制十分复杂。此外,还认为指令系统愈丰富,编译器愈好写,而且编译的效率愈高。③当时主存储器价格较贵,存储器容量有限,因而把存储效率作为衡量处理机体系结构好坏的重要标准。这样,在处理机中大量采用存储效率较高的存储器-存储器操作指令。这种指令要求微操作的数量较大,而且后来证明它是实际执行效率较低的指令类型。还有,为了提高存储效率,采用了可变字长指令字,使指令系统格式更复杂。另外,还采用各种复杂的指令寻址方式,从而使指令系统更为复杂而且非规格化。

但是,CISC 与 RISC 实际上反映的是设计思想。具体实现时这两者的界线并不一定十分明确,如在现代 RISC 产品中仍设置一些复杂指令,这些复杂指令的执行控制依旧采用微程序技术,如 Intel 80960 CA。此外,一些 CISC 处理机也采用了 RISC 设计思想,以减少执行一条指令所需的时钟周期数,如 Intel 80486;或者逐渐向 RISC 产品过渡,如 Intel Pentium 已采用了 RISC 范畴的超标量结构、多流水结构和加强的编译优化技术等。

(李三立)

G

gailǔ suanfǎ

概率算法 (probabilistic algorithm) 一类带有随机操作的算法。又称随机算法。算法在计算的某一步或某些步产生符合规定要求的随机数,然后根据产生出的随机数决定下一步的计算。例如,在计算的某一步有两种选择:执行 A 或执行 B 。此时随机产生一个 0 或 1。若产生的是 0 则执行 A ,若产生的是 1 则执行 B 。这相当于根据掷一枚硬币的结果(正面或反面)决定下一步的计算。

将概率的思想用到算法中始于数值计算,在计算方法中通常称作蒙特卡罗法,是在 20 世纪 40 年代中叶提出的。它的基本思想是建立概率模型,通过统计模拟或抽样得到问题的近似解。通常要求计算结果的期望值等于问题的精确解,并且计算误差的期望值随可供使用的时间增加而减小。近 20 年来概率算法在非数值计算中得到很好的应用。例如,已经设计出关于排序和搜索、素数判定、有限域上的多项式分解和求根、字符串的模式匹配等方面的有效概率算法。概率算法同样也应用到并行计算中,得到概率并行算法。

M. O. Rabin 在 1976 年提出一个判定素数的概率算法,其理论根据是:当 n 是合数时,在 1 到 $n-1$ 的整数中有一半以上是 n 为合数的“见证人”。算法的基本做法是:随机地产生一个 1 与 $n-1$ 之间的整数 b ,检查 b 是否是 n 为合数的“见证人”。若 b 是“见证人”,则计算结束,并得出 n 为合数的结论;否则重复这个过程。至多进行 k 次,若产生的 k 个随机数 b 都不是 n 为合数的“见证人”,则得出 n 为素数的结论。算法所需要的时间为 $O(\log^3 n)$ 。当计算的结果是 n 为合数时,结果肯定是正确的。但是,“ n 为素数”的结果有可能是错误的。此时 n 为合数的概率,即得出错误结果的概率不超过 $1/2^k$ 。当 k 足够大时,这是一个很小的数。譬如,取 $k=10$,错误的概率小于 0.001。这已经是在实验中不大可能发生的事件了。实验表明,算法在实际使用中几乎不会给出错误的结论。

这类概率算法对每一个实例的计算时间是确定的,计算结果服从某种概率分布。也就是说,不能保

证算法给出的解总是正确的,只能保证得出非正确解的概率足够小。还有一类概率算法给出的解总是正确的,而执行时间服从某种概率分布。算法执行时间的期望值通常是输入规模的多项式,但不能排除运行很长时间的可能性甚至有可能给不出解。当然,出现这种坏情况的概率很小。

概率算法通常能大大地提高效率。但有迹象表明,对于 NP 完全问题很可能不存在有效的概率算法。用计算机执行概率算法时,要用伪随机数代替随机数。大多数程序设计语言都有伪语言随机数发生器,用来产生伪随机数。

参考文献

Brassard G, Bratley P. Algorithmics: Theory and Practice. Englewood Cliffs: Prentice Hall, 1988

(张立昂)

gailǔ zìdòngjī

概率自动机 (probabilistic automaton) 一种其结构和所处环境具有随机因素的自动机。又称随机自动机。对于概率自动机,要给出它的初始分布,即开始时内部状态的概率分布;还要规定在某种条件下,概率自动机下一个动作的条件概率。通常,概率自动机的执行结果随其结构不同给出某种概率。概率自动机在信息论、容错计算、学习模型和控制模型等方面都有广泛应用。

概率自动机包括概率时序机,概率有限自动机,概率下推自动机和概率图灵机,主要研究它们的功能、结构、化简和其他性质。概率文法从语言的角度给出了另一研究方向。

概率时序机 时序机的一种推广,它刻画的功能是在给定的初始分布 π 下,输入符号序列 u 后产生输出符号序列 v 的概率 $P_\pi(v|u)$ 。形式上

$$M = (X, Y, Q, P, \pi)$$

其中 X 和 Y 分别是有限的输入和输出符号集; $Q = \{q_1, q_2, \dots, q_n\}$ 是有限状态集; $\pi = [\pi_1, \pi_2, \dots, \pi_n]$ 是初始分布, π_i 表示开始时 M 处于状态 q_i 的概率,满足

$$\pi_i \geq 0, 1 \leq i \leq n, \text{ 且 } \sum_{i=1}^n \pi_i = 1; P \text{ 是条件概率}$$

$P(y, q' | q, x)$, $x \in X$, $y \in Y$, $q, q' \in Q$
表示 M 处于状态 q , 输入符号 x 时, 输出符号为 y 且下一状态为 q' 的概率, 满足 $P(y, q' | q, x) \geq 0$ 且 $\sum_{y \in Y} \sum_{q' \in Q} P(y, q' | q, x) = 1$ 。设 $P(y | x)$ 表示以 $P(y, q_j | q_i, x)$ 为元素的 $n \times n$ 矩阵, η 是各分量都是 1 的 n 维列向量, 则 $P_\pi(y_1 y_2 \cdots y_k | x_1 x_2 \cdots x_k) = \pi P(y_1 | x_1) P(y_2 | x_2) \cdots P(y_k | x_k) \eta$ 。例如, 设 $X = \{0, 1\}$, $Y = \{a, b\}$, $Q = \{q_1, q_2\}$, $\pi = \begin{bmatrix} \frac{1}{4} & \frac{3}{4} \end{bmatrix}$, 且 P :

$$P(a | 0) = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}, \quad P(b | 0) = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} \\ \frac{1}{2} & 0 \end{bmatrix}$$

$$P(a | 1) = \begin{bmatrix} 0 & \frac{1}{2} \\ 0 & 0 \end{bmatrix}, \quad P(b | 1) = \begin{bmatrix} \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

$$\text{则 } P_\pi(a | 0) = \pi P(a | 0) \eta = \begin{bmatrix} \frac{1}{4} & \frac{3}{4} \end{bmatrix} \times$$

$$\begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{2}, \quad P_\pi(ab | 00) = \pi P(a | 0)$$

$$P(b | 0) \eta = \begin{bmatrix} \frac{1}{4} & \frac{3}{4} \end{bmatrix} \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \frac{1}{4} & \frac{1}{4} \\ \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{4}$$

设 π 和 π' 分别为 $\pi_i = 1$ 和 $\pi_j = 1$ 的两个初始分布。如果对于任意的正整数 m 和所有的 $x_k \in X$, $y_k \in Y$, $1 \leq k \leq m$, $P_\pi(y_1 y_2 \cdots y_m | x_1 x_2 \cdots x_m) = P_{\pi'}(y_1 y_2 \cdots y_m | x_1 x_2 \cdots x_m)$ 总成立, 则称状态 q_i 和 q_j 等价。利用状态等价的概念, 可以化简概率时序机。研究状态等价的充分必要条件, 寻求给定概率时序机的化简形式和化简方法, 是重要的研究内容。

概率有限自动机 有限自动机的推广, 它刻画的功能是在给定的初始分布 π 下, 输入符号序列 u 后达到给定状态集 F 中的状态的概率 $P_\pi(u)$ 。形式上

$$M = (X, Q, P, \pi, F)$$

其中 X 和 $Q = \{q_1, q_2, \cdots, q_n\}$ 分别是有限输入符号集和状态集; π 是初始分布; F 是给定的状态集, $F \subseteq Q$; P 是条件概率

$$P(q' | q, x), \quad x \in X, \quad q, q' \in Q$$

表示 M 处于状态 q , 输入符号 x 时, 下一状态为 q' 的

概率, 满足 $P(q' | q, x) \geq 0$ 且 $\sum_{q' \in Q} P(q' | q, x) = 1$ 。如

果 η^F 是 n 维列向量, 它相应于 F 中状态的分量为 1, 其余分量为 0, 则 M 在初始分布 π 下, 输入符号序列 $x_1 x_2 \cdots x_m$ 后达到 F 中状态的概率 $P_\pi(x_1 x_2 \cdots x_m) = \pi P(x_1) P(x_2) \cdots P(x_m) \eta^F$, 其中 $P(x)$ 表示以 $P(q_j | q_i, x)$ 为元素的 $n \times n$ 矩阵。给定一个阈值 λ , $0 \leq \lambda < 1$, 则 M 以切断点 λ 所识别的随机语言

$$T(M, \lambda) = \{u : u \in X^*, P_\pi(u) > \lambda\}$$

通常, $P(q' | q, x) < 1$ 。于是, 随着输入符号序列 u 的长度增加, $P_\pi(u)$ 趋于 0。这样, $\lambda = 0$ 就成为人们最感兴趣的切断点。当 π_i , $1 \leq i \leq n$, 和 $P(q' | q, x)$ 仅取值 1 或 0 时, 概率有限自动机蜕化为有限自动机, 所以随机语言类包含正则语言类。随机语言类的结构, 对运算的封闭性, 概率有限自动机的各种模型及其相互关系, 都是重要的研究课题。

概率下推自动机 下推自动机的一种推广, 它刻画的功能是在给定的状态和下推符号对 (q, Z) 的初始分布 $h(q, Z)$ 下, 输入符号序列 u 后产生输出序列 v 且达到状态 q' 的概率。形式上

$$M = (X, Y, W, Q, P, h, g)$$

其中 X, Y 和 Q 分别是有限的输入、输出符号集和状态集; $h = h(q, Z)$ 是 $Q \times W$ 到 $[0, 1]$ 内的函数, 满足 $\sum_{q \in Q} \sum_{Z \in W} h(q, Z) \leq 1$; $g = g(q)$ 是 Q 到 $[0, 1]$ 内的函数; P 是条件概率

$$P(q', z, v | x, Z, q), \quad q, q' \in Q, \quad x \in X, \quad Z \in W, \quad z \in W^*, \quad v \in Y^*$$

表示 M 处于状态 q , 栈顶符号为 Z , 输入符号为 x 时, 下一状态为 q' , 栈顶符号改写为 z 且输出为 v 的概率, 满足 $P(q', z, v | x, Z, q) \geq 0$,

$$\sum_{q' \in Q} \sum_{z \in W^*} \sum_{v \in Y^*} P(q', z, v | x, Z, q) \leq 1 \text{ 且 } \{P : P(q', z, v | x, Z, q) \neq 0, z \in W^*, v \in Y^*\} \text{ 为有限集。}$$

概率下推自动机简记为 PPA。设 $\Gamma^M = Q \times W^* \times X^* \times Y^*$, 定义 Γ^M 到 Γ^M 的随机函数 $P_k^M(\beta | \alpha)$:

$$P_1^M(\beta | \alpha) = \begin{cases} P(q', z_0, v_0 | x, Z, q), & \text{如果 } \alpha = (q, Zz, xu, v), \\ & \text{且 } \beta = (q', z_0 z, u, vv_0) \\ 0, & \text{否则} \end{cases}$$

$$P_{k+1}^M(\beta | \alpha) = \sum_{\gamma \in \Gamma^M} P_k^M(\beta | \gamma) P_1^M(\gamma | \alpha), \quad k \geq 1$$

这样, 输入符号序列 u 后产生输出符号序列 v 的概率是

$$G^M(v | u) = \sum_{q, q' \in Q} \sum_{Z \in W} \sum_{z \in W^*} h(q, Z)$$

$g(q')P_k^M(q', z, \varepsilon, v \mid q, Z, u, \varepsilon)$, 其中 $k = \lg(u)$ 是 u 的长度。

如果 PPA $M = (X, Y, W, Q, P, h, g)$ 满足 (1) 对于任意的 $x \in X, Z \in W$ 和 $q \in Q$, $\sum_{q' \in Q} \sum_{z \in W^*} \sum_{v \in Y^*} P(q', z, v \mid x, Z, q) = 1$, (2) $\sum_{q \in Q} \times \sum_{Z \in W} h(q, Z) = 1$ 和 (3) 对于任意的 $\alpha = (q, Z, u, \varepsilon), q \in Q, Z \in W, u \in X^*$, $\sum_{\beta \in T^M} P_k^M(\beta \mid \alpha) = 1$, 其中 $k = \lg(u)$, 则称 M 为完全的 PPA。可以证明, 对于任意的 PPA M , 存在完全的 PPA M' , 使得 $G^M = G^{M'}$ 。

PPA M 接受的语言为 φ^M : $\varphi^M(u) = \sum_{v \in Y^*} G^M(v \mid u)$ 。

如果 f 是 PPA M 接受的语言, 则有完全的 PPA $M' = (X, Y, W, Q, P, h, g)$ 接受 f : $f = \varphi^{M'}$, 满足 $Y = \{y\}$ 且 $P(q', z, v \mid x, Z, q) \neq 0$ 隐含 $v = y$ 。这时, M 可以表示为一个六元组 $A = (X, W, Q, P', h, g)$, 其中 $P'(q', z \mid x, Z, q) = P(q', z, y \mid x, Z, q)$ 。 A 被称为概率下推接受器。还可以定义带有切断点 λ 的 PPA M 接受的语言 $L(\varphi^M, \lambda) = \{u: u \in X^*, \varphi^M(u) > \lambda\}, 0 \leq \lambda < 1$ 。带切断点 λ 的 PPA 接受的语言类真包含随机语言类和上下文无关语言类。

概率下推自动机理论研究 PPA 的各种限制类型和接受的语言类, 以及这些语言类对运算的封闭性和相互关系等问题。

概率图灵机 图灵机的推广。形式上

$$M = (W, Q, P, \pi)$$

其中 Q 为有限状态集; W 是带符号集; π 是初始分布; P 是条件概率

$$P(y, q' \mid q, x), \quad q, q' \in Q, \quad x \in W, \quad y \in V, \quad V = W \cup \{R, L, T\},$$

表示 M 处于状态 q , 读写头注视带符号 x 时, M 的“下一动作”的概率, 满足 $P(y, q' \mid q, x) \geq 0$ 且 $\sum_{y \in V} \sum_{q' \in Q} P(y, q' \mid q, x) = 1$ 。“下一动作”为下列四者之一: ① $y = R$: 读写头右移一格, 状态转移到 q' ; ② $y = L$: 读写头左移一格, 状态转移到 q' ; ③ $y \in W$: 读写头不动, 将注视的带符号 x 改写为 y ; ④ $y = T$: 读写头不动, 机器停止。

用概率图灵机可以定义可计算随机函数和带有阈值 λ 的可计算随机函数。可计算随机函数类对函数代入、原始递归、求极小等运算都是封闭的。限制在普通函数类的范围内, 部分可计算随机函数中的普通函数, 恰好是部分递归函数。带有阈值 λ 的可计算随机函数类是不可数的。

概率文法 短语结构文法的推广, 它刻画的功能是给定生成式的初始分布和确定生成式先后使用的概率后, 短语结构文法生成的终结符号序列的概率。形式上

$$G_P = (G, P, \pi)$$

其中 $G = (N, \Sigma, \{f_1, f_2, \dots, f_n\}, X_0)$ 是短语结构文法; $\pi = [\pi_1 \pi_2 \dots \pi_n]$ 是生成式的初始分布; P 是条件概率

$$P(f_k \mid f_j), \quad 1 \leq j, \quad k \leq n$$

表示 G_P 使用生成式 f_j 后, 下一个使用的生成式为 f_k

的概率, 满足 $P(f_k \mid f_j) \geq 0$ 且 $\sum_{k=1}^n P(f_k \mid f_j) = 1$ 。如果 G 是 i 型文法, 则称 G_P 是 i 型概率文法, $i = 0, 1, 2, 3$ 。设 G_P 的派生

$$D: X_0 = u_0 \Rightarrow_{f_{j(1)}} u_1 \Rightarrow_{f_{j(2)}} u_2 \Rightarrow \dots \Rightarrow_{f_{j(r)}} u_r, \\ u_r \in \Sigma^*$$

则派生 D 的概率 $P(D) = \pi_{j(1)} P(f_{j(2)} \mid f_{j(1)}) \dots P(f_{j(r)} \mid f_{j(r-1)})$ 。如果给定阈值 $\lambda, 0 \leq \lambda < 1$, 则带有切断点 λ 的概率文法 G_P 生成的语言 $L_s(G_P, \lambda) =$

$\{u: u \in \Sigma^*, \sum_D P(D) > \lambda, D \text{ 为生成 } u \text{ 的派生}\}$, 和 $L_m(G_P, \lambda) = \{u: u \in \Sigma^*, \text{存在生成 } u \text{ 的派生 } D, \text{使得 } P(D) > \lambda\}$ 。例如, 2 型概率文法 $G_P = (\{X_0, X\}, \{a, b, c\}, \{f_1, f_2, f_3, f_4, f_5\}, X_0), P, [1 \ 0 \ 0 \ 0 \ 0])$, 其中 $f_j, j = 1, 2, 3, 4, 5$, 依次是 $X_0 \rightarrow X_0 X, X_0 \rightarrow a X_0 b, X \rightarrow c X, X_0 \rightarrow ab, X \rightarrow c$; $P: [P(f_1 \mid f_j) P(f_2 \mid f_j) \dots P(f_5 \mid f_j)], j = 1, 2, 3, 4, 5$, 依次是 $[0 \ 1 \ 2 \ 0 \ 1 \ 2 \ 0], [0 \ 0 \ 1 \ 0 \ 0], [0 \ 1 \ 2 \ 0 \ 1 \ 2 \ 0], [0 \ 0 \ 0 \ 0 \ 1], [0 \ 0 \ 0 \ 0 \ 1]$ 。则 $L_s(G_P, 0) = L_m(G_P, 0) = \{a^l b^l c^l: l \geq 1\}$ 。

如果记 $P \cdot i \cdot S = \{L_s(G_P, \lambda): G_P \text{ 是 } i \text{ 型概率文法}, 0 \leq \lambda < 1\}$ 和 $P \cdot i \cdot m = \{L_m(G_P, \lambda): G_P \text{ 是 } i \text{ 型概率文法}, 0 \leq \lambda < 1\}$, 则有 $P \cdot 3 \cdot S = P \cdot 3 \cdot m =$ 正规语言类。在 i 型概率文法 $G_P = (G, P, \pi)$ 中, π 代之以具有非负分量的 n 维向量 δ, P 代之以 $\varphi: \varphi(f_k \mid f_j) \geq 0, 1 \leq j, k \leq n$, 则称 $G_w = (G, \varphi, \delta)$ 为 i 型加权文法。可以证明, 随机语言类 $P \cdot T = \{T(M, \lambda): M \text{ 是概率有限自动机}, 0 \leq \lambda < 1\} \subseteq \{L_s(G_w, \eta): G_w \text{ 是 } 3 \text{ 型加权文法}, \eta \geq 0\}$ 。反之, $\{L_s(G_w, \eta): G_w \text{ 是 } 3 \text{ 型加权文法}, \text{且不含形如 } X \rightarrow Y \text{ 的生成式}, X \text{ 和 } Y \text{ 是变量}; \eta \geq 0\} \subseteq P \cdot T$ 。研究概率文法和加权文法的层次结构及生成的语言类之间的相互关系,

各语言类对运算的封闭性及判定问题等都是重要的研究方向。

参考文献

1. Paz A. Introduction to Probabilistic Automata. New York: Academic Press, 1971
2. Santos E S. Probabilistic Pushdown Automata. Journal of Cybernetics. 1976, 6: 173 ~ 187
3. Salomaa A. Probabilistic and Weighted Gram-

mars. Information and Control. 1969, 15: 529 ~ 544

(郭清泉)

ganxian zixitong

干线子系统 (backbone subsystem) 结构化布线系统中连接各管理间、设备间的子系统。干线子系统所处的位置如图 1 所示。

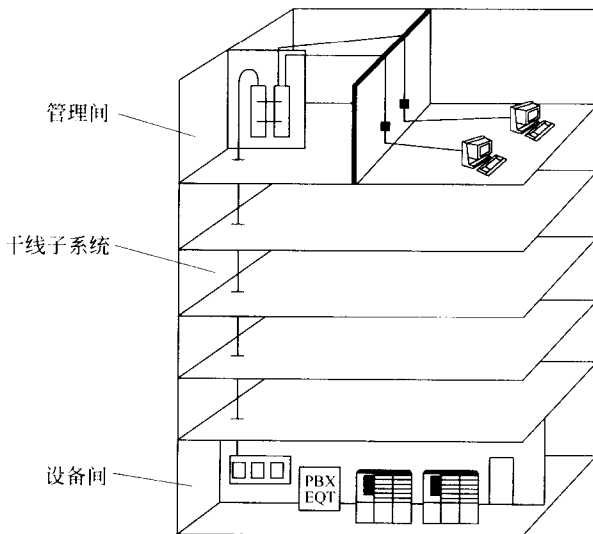


图 1 干线子系统

干线系统也称为垂直竖井系统,它是结构化布线系统的骨干。这个子系统包括:

①供干线电缆走线用的垂直或水平通道;②设备间与网络接口之间的连接电缆;③设备间与建筑群子系统之间的连接电缆;④干线接线间与各卫星接线间之间的连接电缆;⑤主设备间与计算机中心之间的电缆。

结构化布线系统的干线可根据距离的远近和用户对传输速率及传输质量的要求,选择多对数**双绞线电缆**或**光缆**。一般在楼内的语音(言语)通信采用三类的大对数双绞线作为主干;数据主干即可以采用高品质的五类双绞线,也可以采用**光缆**;如果电磁干扰严重,则推荐采用**光缆**作为数据主干。

在设计干线子系统时,首先要确定每一层楼干线需求,再总结出整座楼的干线总体需求,确定干线电缆的种类及大小尺寸,然后确定干线电缆路由通道。

干线的路由通道有两大类,即封闭型和开放型。开放型通道通常是指从建筑物的地下室到楼顶的一个开放空间,中间没有任何楼板隔开,就如同通风道或电梯通道。这种通道给布线施工带来很大的麻烦,所以一般不采用。

封闭型通道是指一连串上下对齐的接线间,每层楼有一间,电缆利用电缆孔或电缆井穿过这些接线间的地板。

电缆孔是很短的管道,通常是用刚性金属管做成。它们嵌在混凝土地板中,这是在浇注混凝土时嵌入的,比地板表面高出 2cm 至 10cm。电缆往往捆在钢绳上,钢绳固定在接线间的墙上,如图 2 所示。

当布线系统的接线间上下对齐,且主干线缆不多的时候,干线子系统布线一般采用电缆孔方式。

电缆井方式是指在每层楼板上开出一些方孔,使得电缆可以穿过这些电缆井从这层楼延伸到别的楼层上去,如图 3 所示。

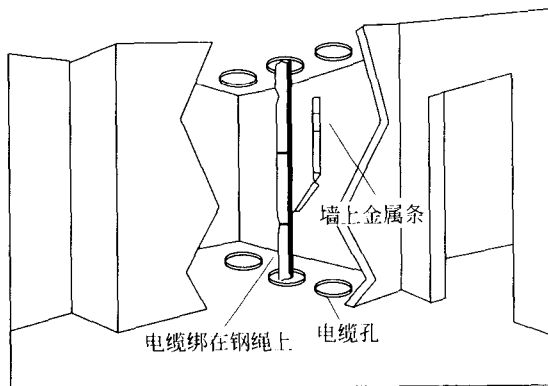


图2 电缆孔方法

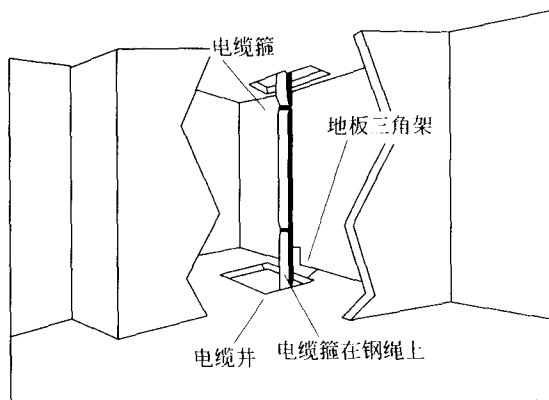


图3 电缆井方法

电缆井的大小依所用的主干电缆的类型、数目及大小来确定。采用电缆井方式,主干电缆在铺设时可以固定在钢绳上,也可以固定在专用的金属爬梯上。电缆井可以允许很多电缆通过,且可以让粗细不同的各种电缆以任何组合方式通过。

为了在发生火灾时,将火、烟和有毒气体限制在始发地区,防止扩散到整座建筑物,对于干线系统的通道要考虑防火措施,安装防火门、防火地板和防火墙。

参考文献

胡道元主编. 网络设计师教程. 北京: 清华大学出版社, 2001

(王晓东)

ganzhiji

感知机 (perceptron) 一种模仿生物感知机制、用于模式识别的简单神经网络。它是美国学者 F. Rosenblatt 于 1956 年提出的,由 S(传感单元)层、

A(联想单元)层和 R(响应单元或称输出单元)三层神经元组成(见图1)。可以学习的只是 A 层与 R 层间的连接权值,所以实际上是只有一层计算单元的神经网络。而且同一层内的神经元之间没有相互连接,不同层之间也没有反馈,所以常称之为单层前向神经网络。

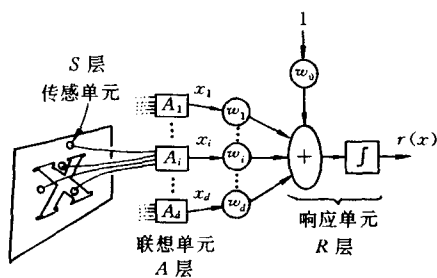


图1 感知机的基本构成

每一联想单元的输出是传感单元信号的某种固定的线性组合,分别记为 x_1, x_2, \dots, x_d , 它们经可调权值 w_1, w_2, \dots, w_d 后送到响应单元 R ; 因此 R 的输入是 $w_0 + \sum_{i=1}^d w_i x_i = w_0 + \mathbf{W}^T \mathbf{X}$, 通常 R 是一个线性阈值单元, 其输出为

$$r(x) = \begin{cases} 1 & w_0 + \mathbf{W}^T \mathbf{X} \geq 0 \\ 0 & w_0 + \mathbf{W}^T \mathbf{X} < 0 \end{cases}$$

依据 $r(x)$ 的取值可将输入模式分类。

这种网络的学习方法是先用已知类别的样本(训练集)对网络进行训练, 具体过程如下述。

设在 t 时刻输入训练样本 $X_t = (x_{s1}, x_{s2}, \dots, x_{sd})$, 此时网络输出为 $r_s(x)$, 对应 X_t 的正确输出已知为 y_s , 记 $\varepsilon_s = y_s - r_s(x)$ 为网络输出与正确输出 y_s 的误差, 则 w_i 按如下方式修正:

$$\begin{aligned} w_i(t+1) &= w_i(t) + \varepsilon_s x_{si} \\ &= w_i(t) + [y_s - r_s(x)] x_{si} \end{aligned}$$

以上学习算法称为感知机学习算法, 可以证明, 当输入模式是线性可分时, 该算法可在有限步骤内收敛, 且训练后的网络可对所有输入模式正确分类。也可以用使均方误差达最小的学习算法, 当模式不是线性可分时它收敛于使均方误差最小的解。

感知机的输出所形成的判别函数是一个超平面, 所以原则上它只能解决线性可分的模式分类问题, 对于分界面不是超平面的模式(非线性可分), 用感知机学习算法时不论怎么调整权值, 都不能实现正确分类。例如, 对两个输入一个输出的异或运算[输入(0,1)或(1,0)时输出应为1, 输入(0,0)或(1,1)时输出应为0]问题, 感知机就不能胜任。对于感知机的能力和限度, Minsky 和 Papert 在他们的著作“Perceptron”中有很详细的分析。

参考文献

1. 史忠植. 神经计算. 北京: 电子工业出版社, 1993
2. Васильев В И 著. 机器识别方法与系统. 边肇祺, 阎平凡译. 北京: 科学出版社, 1991
3. Minsky M, Papert S. Perceptrons. MIT Press, 1988
(阎平凡)

gaoci daishu fangcheng jiefa

高次代数方程解法 (solution of polynomial equation) 根据高次代数方程的特点, 研究如何采用数值计算方法来求 n 次多项式方程

$P_n(x) \equiv a_0 x^n + a_1 x^{n-1} + \dots + a_{n-1} x + a_n = 0 \quad (1)$
根的(即 $P_n(x)$ 的零点)方法与过程。式(1)中 a_0, a_1, \dots, a_n 为实数或复数, 且 $a_0 \neq 0, n \geq 2$ 的正整数。由代数方程基本定理, 在复数域内 n 次代数方程有 n 个根 x_1, \dots, x_n , 其中可能有相同的根, 称为重根。对二次方程 $a_0 x^2 + a_1 x + a_2 = 0, a_0 \neq 0$, 它的两个根可表示为 $x_1, x_2 = \frac{-a_1 \pm \sqrt{a_1^2 - 4a_0 a_2}}{2a_0}$ 。三次和四

次方程的根也可用根式表示, 但这些公式相当复杂, 不便于计算。五次以上的代数方程的根已不可能用根式表示, 因此对三次以上的代数方程求根, 一般都用数值解法。由于高次方程是非线性方程, $f(x) = 0$ 左端换成多项式 $P_n(x)$, 因此, 求非线性代数方程的各种数值解法, 如牛顿法及其他迭代法均是高次方程的有效解法(参看非线性代数方程组数值解法)。但也还有针对高次方程特点的一些特殊解法, 较重要的有劈因子法, 伯努利法, 卢斯表格法和圆盘迭代法。

劈因子法用来求多项式 $P_n(x)$ 的一个二次因子, 然后通过解二次方程求得高次方程(1)的一对复根或两个实根。用一个近似的二次因子 $w(x) = x^2 + ux + v$ 除 $P_n(x)$, 得商 $q(x)$ 及余式 $r(x) = r_1 x + r_2$, r_1 及 r_2 由 $w(x)$ 的系数 u, v 确定, 于是有

$$P_n(x) = w(x)q(x) + r(x)$$

若 $r_1 = r_2 = 0$, 则 $w(x)$ 就是 $P_n(x)$ 的一个二次因子。否则, 可令 $w^*(x) = x^2 + (u + \Delta u)x + (v + \Delta v)$, 用 $w^*(x)$ 除 $P_n(x)$, 得余式 $r^*(x) = (r_1 + \Delta r_1)x + (r_2 + \Delta r_2)$, 要求 $r^*(x) = 0$, 即 $\Delta r_1, \Delta r_2$ 满足方程 $r_1 + \Delta r_1 = 0, r_2 + \Delta r_2 = 0$, 由此可得到修正量 Δu 及 Δv 的联立方程

$$\begin{cases} \frac{\partial r_1}{\partial u} \Delta u + \frac{\partial r_1}{\partial v} \Delta v + r_1 = 0 \\ \frac{\partial r_2}{\partial u} \Delta u + \frac{\partial r_2}{\partial v} \Delta v + r_2 = 0 \end{cases}$$

利用已知关系可求出方程系数 $\frac{\partial r_1}{\partial u}, \frac{\partial r_1}{\partial v}, \frac{\partial r_2}{\partial u}, \frac{\partial r_2}{\partial v}$, 从而可求得 Δu 及 Δv 得到 $w^*(x)$, 它是比 $w(x)$ 更好的二次因子。重复以上过程直到求出收敛的二次因子为止, 最后再求二次因子 $w^*(x)$ 的一对根。此法不用复数运算就可求复根, 这是该方法的优点。

伯努利法可用来求方程模最大或最小的根。卢斯表格法可用于对方程复根的隔离和判断方程在某区域根的个数, 并可求出最大实部根。圆盘迭代在根隔离后可求出全部根, 并具有收敛速度快和并行

计算的优点。

在高次代数方程求根过程中,往往会遇到病态多项式,其系数的微小变化会引起零点的很大变化。因此,在计算机编程求根时,通常应采用双精度计算。另外,求 $P_n(x)$ 全部零点时,如用降阶办法,求根次序应按根模由小到大次序进行,才不会影响后面求根的精度。

代数方程求根是科学与工程计算中最基本的问题之一,在系统稳定性研究和代数特征值问题计算中经常用到。

参考文献

清华大学,北京大学《计算方法》编写组. 计算方法(上册). 北京: 科学出版社, 1974 (李庆扬)

gaoji yuyan

高级语言 (high level language) 不反映特定计算机体系结构的程序设计语言。它的表示方法要比低级语言更接近于待解问题的表示方法。其特点是在一定程度上与具体机器无关,易学、易用、易维护。当高级语言程序翻译成等价的低级语言程序时,一般说来,一个高级语言语句要对应多条机器指令,相应编译程序所产生的目标程序往往功效较低。

1952 年瑞士数学家 H. Rutishauser 首先提出高级语言的概念,第一个实用高级语言是美国 IBM 公司 J. Backus 等人于 1956 年研制的 FORTRAN, 1960 年相继公布了一种用于事务处理的语言 COBOL 和算法语言 ALGOL 60。这三种语言经过多次修改,出现多个新的标准文本,至今仍有一些领域内占有一定的地位。随后,有人试图研制一类兼收并蓄多种程序设计语言功能的大型通用语言。例如,IBM 公司设计了汇集型语言 PL/1, 希望它取代 FORTRAN 和 COBOL, 因此只需支持一个语言。又如,可扩充语言 ALGOL 68 具有很强的上下文有关描述能力。但因这类语言过于复杂,系统规模大,效率低,未能流行。结构化程序设计出现以来,人们已设计出多种符合结构化程序设计的语言,如 PASCAL, Ada 等。

随着计算机应用领域的扩大,产生了函数式、逻辑、面向对象程序设计语言,例如: FP, PROLOG, Smalltalk 等等(参见程序设计语言)。

高级语言种类千差万别,但是,一般说来基本成分有四种。①数据成分:用以描述程序中所涉及的数据;②运算成分:用以描述程序中所包含的运算,例如:表达式;③控制成分:用以表达程序中的控制

构造,例如:条件语句;④传输成分:用以表达程序中数据的传输,例如:输入输出语句。

高级语言正向模块化、简明性、形式化、并行化和可视化等方向发展。

参考文献

1. Horowitz E. Fundamentals of Programming Language. Rockville: Computer Science Press, 1983

2. Sammet J E. Programming Language: History and Fundamentals. Englewood Cliffs: Prentice Hall, 1969

(郑国梁 徐宝文)

gaojie luoji

高阶逻辑 (higher-order logic) 将一阶逻辑的阶数推广至高阶的逻辑。一阶逻辑中的谓词、函词称为一阶谓词、一阶函词,其变元符号都是个体变元符号,量词仅作用于个体变元,这限制了一阶逻辑的表达能力。去掉对变元、谓词、函词的上述限制,就发展成为高阶逻辑。高阶逻辑与类型论有紧密联系。

可以从多方面来推广一阶逻辑。如允许量词不仅可用来约束个体变元,而且也可用来约束谓词变元及函词变元,即允许量词的指导变元不仅可以是个体变元,也可以是谓词变元及函词变元。这样就得到了通常所说的二阶逻辑。例如, $\exists A \exists B (A(x) \rightarrow B(x))$ 便是一个二阶逻辑公式。

如果对含有自由谓词变元,函词变元的公式利用概括原则抽象出一个新谓词,那么这个新谓词就是一个以谓词、函词为变元的二阶谓词。例如,对于含二元谓词 R 的公式。

$$\forall x \forall y (R(x, y) \rightarrow R(y, x)),$$

根据概括原则可抽象出一个表示对称性的谓词 sym , 即

$$\text{sym}(R) \leftrightarrow \forall x \forall y (R(x, y) \rightarrow R(y, x)).$$

一般地,由含有至高为 n 阶的自由谓词变元、函词变元的公式根据概括原则抽象出来的谓词、函词便是 $n+1$ 阶谓词、函词。这样可以逐步得到更高阶的谓词、函词,再添入以各阶谓词变元、函词变元为指导变元的量词,更高阶的逻辑便也逐步得到了。但要注意,二阶逻辑中只是含有一阶谓词、函词(自由的及约束的),含二阶谓词、函词的便是更高阶的逻辑了。依照 A. Church 的划分,如果在系统中最高阶的谓词及函词(设为 n 阶)均是自由的,便叫做 $2n-1$ 阶逻辑,最高阶(设为 n 阶)的谓词及函词有约束的,便叫做 $2n$ 阶逻辑。

高阶逻辑有比一阶逻辑更强的表达能力。例如,一阶逻辑不能表达有穷集合的概念,在高阶逻辑中却可用如下公式表达:

$$\forall f(\forall x \forall y(f(x) = f(y) \rightarrow x = y) \\ \rightarrow \forall x \exists y f(y) = x)$$

因为一个集合是有穷集合当且仅当该集合上的单射必是满射,所以该公式解释为真命题当且仅当论域是有穷集。

高阶逻辑失去了一阶逻辑的某些良好性质。逻辑有效式的集合不是递归可枚举集,紧致性定理也不再成立,因此不存在可靠且完全的公理系统。基数定理也不成立。

集合论可以表述全部数学,但往往表述得十分复杂难懂,用高阶逻辑表达要简单明了得多。例如,在集合论中用十分冗长的公式才能表述有穷集合的概念。在集合论中对于各阶函数、关系等不加区别,组成一个共通的域,因此对概括原则的使用必须施加适当限制,否则极易产生悖论。而在高阶逻辑中,可以无条件地使用概括原则而不致引起悖论。高阶逻辑对于证明论、递归论(特别是计算复杂性)、公理集合论和一阶逻辑中的证明的研究都是很有意义的。

参考文献

Monk J D. Mathematical Logic. New York; Springer-Verlag, 1976

(何自强 王戟)

gaosu huanchong cunchuqi

高速缓冲存储器 (cache) 位于中央处理器与主存储器之间,对程序员透明的一种高速小容量存储器。简称**高速缓存**。

高速缓冲存储器简称高速缓存。它是存储器层次结构的最顶层,其下层是容量相对较大和访问时间较长的主存储器。在配备有高速缓存的计算机中,每次访问存储器都先访问高速缓存,若欲访问的数据在高速缓存中,则访问到此为止;否则,再访问主存储器,并把有关数据取入高速缓存。这样,如果大部分针对高速缓存的访问都能成功,则在保持主存储器容量不变的前提下,访存速度可接近高速缓存的存取速度。高速缓存的设置是所有现代计算机系统发挥高性能的重要因素之一。

高速缓存的工作机制体现了**局部性原则**。所谓局部性,是指程序访问代码和数据的不均匀性,它包括:①时间局部性:如果某位置已被访问,则该位置很可能在短时间内还要再被访问;②空间局部性:

如果某位置已被访问,则其邻近位置很可能还要被访问。因此,只要程序有较好的访存局部性,高速缓存就能发挥作用。

命中率

高速缓存的组织及对高速缓存的访问都是以行或块为单位的,通常1行包括1个或多个字。行也是高速缓存与主存储器交换数据的最小单位。访存时,若能在高速缓存中找到所需数据,称为高速缓存命中,否则就是不命中。命中次数与访存总次数的比率称为命中率,而不命中次数与访存总次数的比率称为不命中率。显然,不命中率 = 1 - 命中率。命中时的访问时间(包括确定是否命中的时间)称为命中时间,不命中时,将主存块替换入高速缓存并提供给中央处理器所需的时间称为不命中损耗。不命中时的访存时间是命中时间加上不命中损耗。

命中率是高速缓存性能的一个重要指标,它不仅依赖于硬件实现,而且与应用程序有关。但是,命中率是与硬件速度无关的参数,从而不能单独用于高速缓存的性能评估。较好的标准是平均访存时间:

$$\text{平均访存时间} = \text{命中时间} + \text{不命中损耗} \\ \times \text{不命中率}$$

平均访存时间的单位可以是绝对的(如25 ns),也可以是相对的(如2个时钟周期)。尽管把块变大有助于提高命中率,但同时也增加了不命中时的传送时间,所以块的大小应有利于缩小平均访存时间。

块映射策略

当需要将来自内存的新数据块装入高速缓存时,由块映射策略决定它的存放位置。根据块在高速缓存内的位置,块映射策略可分成如下3类:

(1) 直接映射 每个块在高速缓存中只能有1个位置。该位置通常由求模运算得到,即:

$$\text{高速缓存内块号} = \text{块地址} (\bmod \text{高速缓存中块数})$$

(2) 全相联映射 块可以放在高速缓存中的任意位置。

(3) 组相联映射 高速缓存分为若干个组,块先映射至组,在组中的位置任意。块向组的映射通常也采取取模运算。块地址中对应于组号的若干位称为索引。若组中有 n 块,则称高速缓存是 n 路组相联的。

高速缓存对每个块都设立1个标志域,其中包括块地址和该地址是否有效的信息。为了实现快速访问,在相联映射型高速缓存中,总是同时查找所有的标志。

块替换策略

块替换策略是高速缓存设计的另一个重要方面。当不命中时,必须将相应的主存块取入高速缓存,相应地,要把其中已有的某一块替换出去。若高速缓存内无效块,则替换是不成问题的。对于直接映射高速缓存,只有1个块可供替换。对于相联映射型的高速缓存,主要有随机和最近最少使用(LRU)两种替换策略。LRU策略利用了时间局部性,可能实现较高的命中率,但当需要追踪的块较多时,实现起来比较复杂。

高速缓存的写操作策略

高速缓存的写操作可以采用如下两种策略:

①直写:数据不仅写入高速缓存,同时写入下层的主存储器;②回写:数据只写入高速缓存,只有当相应块被替换时才写回主存。对回写型高速缓存,标记中应维持一个写过位,以区分未写过的块。回写方式有利于提高写的速度,但会增加不命中时间(不命中时必须等被替换的写过块写回主存后,才能从主存将被访问的块取入高速缓存)。直写较易实现,且主存中数据总是与高速缓存中数据一致。为了提高写速度,减少中央处理器的等待时间,直写高速缓存常采用写缓冲技术,即中央处理器在数据写入高速缓存和写缓冲器后,就开始执行后面的指令,而由系统配备单独的控制逻辑将写缓冲区的内容写入主存。

当写数据不命中时,亦可采取两种策略:①取写:先把块取入高速缓存,再重复命中的写过程。回写型高速缓存常用此方案。②绕写:只修改主存内容而不取入高速缓存。直写型高速缓存常用此方案。

高速缓存的性能

当高速缓存不命中时,直接访问主存会增加程序的执行时间。对于执行每条指令的平均周期数(CPI)较小的系统,这种影响的后果很严重。对CPI为1.5的系统(在命中情况下),若不命中率为11%,不命中损耗为10周期,平均每条指令访存1.4次,则实际CPI为3.0,即考虑高速缓存后,执行时间增加了1倍。另外,由于主存访问时间在不同机器中基本相同,在时钟频率较高的系统中,高速缓存不命中带来的时钟周期损失就更大。

高速缓存不命中源于下列3种情形:

(1)首次装入 首次访问某块时,必然导致不命中。增加块内字节数有利于减少此类不命中。

(2)容量不足 高速缓存中不能放下程序运行

所需的所有块。只有增加高速缓存容量才能解决。

(3)组内冲突 映射到同一组的块数超过组内可容纳的块时,发生组内冲突。显然全相联高速缓存中没有这类不命中,但块数较多时,全相联是最昂贵且访问速度最慢的。

由容量不足或组内冲突导致的不命中常常会形成颠簸,即某些块在高速缓存和主存之间频繁传送。增加高速缓存容量和相联性有助于缓解颠簸现象。

为了争取速度,高速缓存总是用最快速的SRAM实现,从而增大容量会导致成本的增长。更重要的是,微处理器芯片的面积有限,不可能将片上高速缓存做得很大。为了弥补片上高速缓存容量的不足,以及高速缓存与主存间的速度差距,可以采用多级高速缓存的方案,即在片上高速缓存与主存间增加1级或多级速度稍慢但容量较大的高速缓存,使整个存储层次的速度变化和容量变化都比较平缓。因为第二级高速缓存的作用主要是用来减少第一级高速缓存的不命中损耗,并消除因容量不足导致的不命中。它的设计可以采用不同于一级(或片上)高速缓存的策略,如采用较高的相联度以提高命中率,使用较长的块来开发空间局部性等。

地址转换

通常,中央处理器给出的访存地址是逻辑地址(或虚地址),它必须由地址转换机构转换成物理地址后才能访问主存。为了减少高速缓存访问的命中时间(它决定中央处理器的时钟周期),许多系统采用了虚地址高速缓存。这类高速缓存中的地址标志是虚地址,中央处理器给出的地址不经转换就可用于高速缓存访问,而同时该地址可由地址转换机构转换成物理地址,从而一旦高速缓存不命中,可立即用实地址访问主存。虚地址高速缓存带来的一个问题是,当进程切换时必须清除高速缓存,因为不同进程的虚地址空间都是一样的。这对系统(尤其是对上下文频繁切换的事务处理系统)性能有一定的影响。

另外,考虑到程序访问指令和数据的不同特点,可以设置专门存放指令和专门存放数据的高速缓存,它们可采用不同的容量和策略。指令和数据分开存放有利于指令流水线的实现。

参考文献

1. Jim Handy. The Cache Memory Book. San Diego: Academic Press Inc., 1993
2. Alan Jay Smith. Cache Memories. ACM Com-

puting Surveys, 1982, 14(3):473~530 (唐志敏)

gaosu huanchong cunchuqi yizhixing

高速缓冲存储器一致性 (cache coherence)

在采用层次结构存储系统的计算机系统中, 保证高速缓冲存储器中数据与主存储器中数据相同的机制。

在单处理机系统中, 高速缓冲存储器数据与主存数据的不一致主要是由输入输出(I/O)操作(如直接存储器存取)引起的。在几乎所有的计算机系统中, I/O数据总是直接从主存进出而不经高速缓冲存储器, 从而, ①在使用回写型高速缓冲存储器的系统中, 输出数据时, I/O系统看到的只是主存中的旧数据(新数据在高速缓冲存储器中); ②输入数据时, 对各种高速缓冲存储器, 中央处理器只能看到其中的旧数据(新数据在主存中)。为了避免这样的数据不一致, 需要硬件或操作系统的干预。例如, 在进行输出操作时, 由操作系统将有关地址的数据移出高速缓冲存储器, 或者由硬件检查高速缓冲存储器中的相应标志, 发现与输出数据地址相同时, 立即写回主存; 在进行输入操作时, 可由操作系统保证输入数据区不被缓冲, 或先将高速缓冲存储器中有关地址的内容清除。

在使用虚地址高速缓冲存储器的系统中, 系统软件将两个不同的虚地址映射到同一物理地址的别名现象会导致高速缓冲存储器数据与主存数据的不一致以及高速缓冲存储器中出现同一数据的两个副本。通过对别名进行简单的限制就可避免这种形式的不一致。

共享存储多处理机的高速缓冲存储器一致性

高速缓冲存储器一致性问题主要出现在共享存储器的多处理机系统中。多处理机对高速缓冲存储器一致性的要求可表述为: 任意一次取数操作得到的结果都必须是最近一次对该数据进行写操作时写入的值。由于互联网的延迟和系统中各种缓冲机制的影响, “最近”的含义实际是无法准确定义的。共享存储器的一致性模型研究的就是从程序员的角度如何容忍对“最近”一词的模糊认识; 而高速缓冲存储器一致性机制则在此基础上讨论如何维持同一数据的多个相同副本。

虽然共享数据的多副本现象是高速缓冲存储器数据不一致的主要原因, 但有时非共享数据也会出现不一致的问题。其原因除了前述的I/O操作外, 还有进程迁移。若一个进程在处理机A上访问了

变量X, 迁移到处理机B后又修改了X, 则当它再回到A时, 就会读到A的高速缓冲存储器中存在着的X的旧值。如果强制每次上下文切换时都清除高速缓冲存储器, 就不会出现这种情况, 但系统性能将会下降。更好的方法是在高速缓冲存储器数据的标志中增加进程标识符(PID)信息。

因多副本导致的高速缓冲存储器不一致可用下例说明(考虑3个处理机的情形):

(1) 处理机A和处理机B分别读变量X(设其初值为12), 则X分别进入A和B的高速缓冲存储器;

(2) 处理机A将X修改为16, 则cache A中X=16, cache B中X=12;

(3) 处理机A, B, C分别读X, 则A得到16, B得到12, C得到的值取决于高速缓冲存储器的写策略: 若采用直写策略, C得到存储器中的新值16, 若采用回写策略, 则得到存储器中的旧值12。

避免出现这一问题的最简单方法是不允许共享数据进入高速缓冲存储器, 如伊利诺依大学研制的CEDAR就采用了这一方式。但这样做对系统性能的影响很大, 尤其是在分布式共享存储器系统中, 不缓存共享数据就无法利用远程访问中的局部性, 从而大大增加了访存的延迟时间。

一致性协议

彻底解决上述问题的途径是采用硬件或软件的机制(协议), 保证当某一个处理器更新高速缓冲存储器中的共享数据时, 主存储器和持有该数据副本的其他高速缓冲存储器能及时知道这一动作。下述通用协议就能实现这一目标。

(1) 读共享数据时, 若不命中, 则先询问所有其他高速缓冲存储器, 以得到最近被写入的数据。

(2) 写共享数据时, 不论是否命中, 都要求每个高速缓冲存储器检查自己是否持有该数据的副本。对于其他高速缓冲存储器中的副本, 可采用如下处理方式之一: ①写使无效: 将所有副本所在行置为无效状态; ②写广播或写更新: 将写入的新值送给所有持副本的高速缓冲存储器。

虽然通用协议能保证高速缓冲存储器数据的一致性, 但其广播特性对系统性能有较大的影响。事实上, 一致性维护很容易使互联网带宽达到饱和。

实际使用的高速缓冲存储器一致性协议可分为3种类型, 即监听协议、目录机制和编译制导的一致性维护。下面对这3种协议分别进行简单的介绍。

(1) 监听协议 主要用于总线结构的多处理

机。每个处理机通过自己的监听硬件监视其他处理机的存储器访问,并检测自己高速缓冲存储器内的数据是否被其他处理机修改。如果发现高速缓冲存储器中数据已被修改,则或者无效或者更新本地数据。在写无效时,下次访问该数据时高速缓冲存储器将不命中,从而处理机可从主存或其他高速缓冲存储器中获得正确的数据。监听机制利用了总线的广播特性,实现较简单,但由于总线带宽的限制,一般只适用于小规模的多处理机系统。也有人考虑将这一协议用于层状总线系统和环状系统。在监听机制中,与一致性有关的信息存放在高速缓冲存储器中,而这种机制有时也称为基于高速缓冲存储器的一致性协议,以区别于基于目录的协议。

多级互联网、二维网格、超立方体和胖树等连接网络具有较总线高得多的带宽,但访问存储器的延迟时间也加长了,因而对局部高速缓冲存储器的性能也有更高的要求。由于在这些网络中处理机已不再具有监视所有存储器操作的能力,监听协议无法适用。硬件目录机制及编译制导机制就是为这类系统设计的。

(2) 基于目录的高速缓冲存储器一致性协议
在这种协议中,任一处理机所持有的数据副本的信息都由目录维护,处理机从目录中可以得知它的动作是否会导致数据的不一致。处理机写数据前,必须从目录处获得对某一数据块的独占访问权。而目录在接到处理机对独占访问权的申请后,先向所有持有该数据块副本的处理机发无效命令,直到收到所有这些处理机的应答信号后才将独占访问权交给申请处理机。当处理机试图读被别的处理机独占的数据块时,它先向目录请求读不命中服务。目录于是向持有独占副本的处理机发命令,要求它将数据写回存储器(同时收回独占访问权)。当收到该数据后,再将它转发给请求读数的处理机。

目录机制的具体实现可能与上述过程有所不同。例如,系统可以选择写更新而不是写无效的策略。不同于监听机制(其中状态总位数正比于所有高速缓冲存储器大小之和),目录机制中需维持的信息正比于系统中的存储器总容量,且每个存储器数据块除状态位外,还必须配备记录该块数据在哪些高速缓冲存储器中有副本的存储位。因此,目录信息的存储位置及存储策略(集中还是分布)是一个需要仔细斟酌的问题。一般可采取全映像或位向量、有限指针、链式目录等方式。另外,在允许弱一致性模型(参见**共享存储**)的系统中,对一般共享数

据(有别于同步变量),可以不等收到所有的无效应答信号就开始写操作,以提高系统的整体性能。

(3) 编译制导的一致性协议 在编译时确定哪些高速缓冲存储器数据块会过时,并生成特殊的指令插入代码段中,以防止处理机使用这些可能过时的数据(通常采用将有关数据清除出高速缓冲存储器的办法)。与目录机制相比,简单的编译制导策略有可能导致过多的高速缓冲存储器数据无效,从而降低存储系统的性能。

性能与成本

多处理机中选择高速缓冲存储器一致性机制的主要考虑是它的性能,即它是否允许高速缓冲存储器有效地减少访存延迟。另一个需要考虑的因素是实现成本,如目录信息的存储量和控制逻辑的复杂度。此外,高速缓冲存储器数据块的大小对一致性机制的性能也有较大的影响。原则上,大的数据块可以充分地利用数据的空间局部性,发挥高速缓冲存储器的预取功能,但也带来了较难处理的**假共享**问题。通常,高速缓冲存储器的数据传送单位是块,从而一致性维护的基本单位也是块。当两个处理机分别频繁地写同一块中的两个字时,一致性机制认为它们共享同一数据块,导致处理机间来回地发无效命令并进行数据传送。

参考文献

1. Kai Hwang. Advanced Computer Architecture: Parallelism, Scalability, Programmability. New York: McGraw-Hill Inc., 1993
2. Jim Handy. The Cache Memory Book. San Diego: Academic Press Inc., 1993
3. David J. Lilja. Cache Coherence in Large-Scale Shared-Memory Multiprocessors: Issues and Comparisons. ACM Computing Surveys, 1993, 25(3): 303 ~ 338 (唐志敏)

gaosu juyuwang

高速局域网 (high speed local area network)

数据传输速率超过 100 Mb/s 可以满足高速、宽频的数据通信和各种网络应用需求的**局域网**。

过去 20 多年,计算机的处理速度提高了百万倍,但网络速率只提高了上千倍,因此网络速率成为整个系统的瓶颈,亟须提高。例如在 1980 年以太网产品刚问世时,个人计算机(PC)的处理能力只有 0.3 MI/s,而以太网速率达 10 Mb/s。而现今 PC 的处理能力已达几千个 MI/s,但以太网的速率却没有

较大的提高。

从用户的需求看,新的应用不断提出,例如分布计算、多媒体应用、企业联网和部门之间的网络互连,这些应用都要求有更高的网络速率和频宽。而计算机处理能力和性能价格比的不断提高,刺激用户开发更复杂的、对频宽要求高的应用,这就要求网络能支持在桌面计算机上运行的基于图像和图形的各种应用。

高速局域网的主要应用包括:高速局域网主干网;高性能计算环境,如基于高性能并行接口(HIP-PI)协议局域网接口的多计算机系统;多媒体桌面系统,如多媒体会议系统;分布计算和 workstation,如**客户-服务器模式**的分布计算。网络性能是分布计算的关键,例如对于 10 亿次计算能力的系统需要每秒几千兆位的网络速率以及延迟很小的通信。还有像可视化计算、实时处理仿真、CAD/CAM、**联机事务处理**等应用都需要高速局域网的支持(参见**科学计算可视化、计算机仿真和以太网**)。

高速局域网不同于传统的局域网,为了解决通信量很大的猝发式通信和已有的系统频带之间的不匹配问题,解决网络堵塞的问题,以及获得宽频、低延迟的网络特性,需要研究新的设计原则和方法。例如:要研究高速局域网的体系结构和高速接口在体系结构方面采用基于**交换**的结构,而不是传统的局域网中基于**介质共享**的结构;研究快速处理网络堵塞的方法,如**链路-链路的流控**;采用单元级(即小分组)的复用方法;采用虚拟连接的体系结构等。高速接口包括和主机的接口、专用系统的接口以及其他网络的接口。

高速局域网的发展方向有以下三点:①满足新的应用需求;②采用先进的技术;③降低软、硬件的价格。

参考文献

胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002

(胡道元)

gaosu shuzi xinhao chuanshu

高速数字信号传输 (high speed digital signal transmission)

在现代计算机和数字设备中由于数字电路速度的提高,信号在互连线上的传输时间可与电路本身的级延迟时间相比拟时所产生的信号传输问题。它主要是研究和解决高速数字信号在传输时发生的反射和串扰两大问题。由于传输线自身特性(特性阻抗、传输速度、损耗等)

的不同,传输线沿途分布的负载的不同,多段传输线接续的不均匀和端接不匹配等复杂因素所引起的反射,使数字信号在传输中产生畸变,表现为边沿成台阶状,或产生上冲和下冲。信号在传输线中传输,通过互感和电容对相邻传输线产生串扰,也使被串扰的信号发生畸变。反射和串扰使信号发生的畸变会增加电路的级延迟,影响系统速度,甚至会使电路产生误动作,影响系统的可靠性。所以研究和掌握反射和串扰的规律,制定相应的工程规范对其进行控制,对于保证系统的速度和稳定可靠是至关重要的。

反射

特性阻抗 在数字电路的应用中往往把传输线看成理想的无损传输线,它的特性阻抗

$$Z_0 = \sqrt{\frac{L_0}{C_0}}$$

其中, L_0 为传输线单位长度的电感, C_0 为传输线单位长度的电容。可以认为 L_0 和 C_0 与频率无关,特性阻抗呈现为与频率无关的纯电阻。

电信号在传输线中以接近于光速的速度传输,经过一定长度的传输线有一定的延迟。经过单位长度传输线的延迟 t_{pd} 是单位长度传输线的电容和电感的乘积的平方根,它由传输线介质的相对介电常数 ϵ_r 决定,即有

$$t_{pd} = \sqrt{L_0 C_0} = 0.333 \times 10^{-8} \sqrt{\epsilon_r} \text{ s/m}$$

常用的传输线有同轴电缆、带状电缆、印制线和双绞线等,它们所用的介质材料决定了相应的传输延迟。同轴电缆的 t_{pd} 为 4 ~ 5 ns/m, Z_0 为 50 Ω , 75 Ω 和 125 Ω 。双绞线的 t_{pd} 为 4.5 ~ 6 ns/m, Z_0 为 50 ~ 100 Ω 。常用的印制电路板板材的 ϵ_r 在 3.0 ~ 6.2 之间, t_{pd} 为 5.78 ~ 8.3 ns/m。

传输线的特性阻抗 Z_0 是一个重要的系统参数。 Z_0 低的系统稳定性较好,但电路的驱动功耗大。通常不同的逻辑电路选择不同的特性阻抗。ECL 电路常选用 $Z_0 = 50 \Omega$, TTL 电路常选用 $Z_0 = 75 \Omega$ 。一旦确定了 Z_0 ,在制造和选择传输线时,要将 Z_0 控制在一定范围内。

多次反射 设用内阻为 R_0 的信号源 $E(t)$ 驱动特性阻抗为 Z_0 的传输线,传输线终端接有阻抗为 Z_L 的负载。传输线始端 S 的入射电压

$$U_{s0}(t) = \frac{Z_0}{Z_0 + R_0} E(t)$$

$U_{s0}(t)$ 经长度为 l 的传输线向终端前进,经过 $T_d = l t_{pd}$ 的延迟到达终点 L。当负载阻抗 Z_L 等于特性阻

抗 Z_0 时,入射电流全流入 Z_L ,终点 L 的端电压等于入射电压 $U_{s0}(t)$,没有反射产生。这是波形完全不畸变传到终点的理想情况。当负载阻抗不等于特性阻抗时,就产生第一次反射,终端电压等于入射电压和反射电压之和。第一次反射电压 U_{Lr1} 沿相反方向又经 T_d 入射到始端 S,由于始端的内阻一般不等于特性阻抗,又产生新的反射电压 U_{Sr1} ,这时始端电压为 $U_{S1} = U_{Lr1} + U_{Sr1}$ 。始端的第一次反射电压 U_{Sr1} 还继续传向终端,再产生第二次反射。这个过程一直继续下去,直到第 n 次反射电压接近零,波形达到稳定为止。始端电压是 $U_{s0}(t)$ 和多次反射后形成的始端电压在时间轴上的迭加,即始端所有入射电压和反射电压的总和。同样,终端电压是终端的多次入射电压和反射电压对时间的迭加。

传输线沿线各点的数字信号是驱动信号和多次反射迭加形成的,反射程度决定了信号畸变的形状和大小。传输线的特性阻抗、传输速度和长度、多段传输线的接续方式和均匀性都直接影响到反射。

匹配终端 数字电路既是驱动电路又是负载电路,它的输出阻抗构成传输线驱动电路的内阻 R_0 ,输入阻抗构成传输线的负载 Z_L 。数字电路(包括 TTL, ECL 和 CMOS 电路)的输入阻抗和输出阻抗都是非线性的。输入阻抗的电阻成分在 0 态和 1 态都呈几十 k Ω 的大电阻,在开关过渡区则在百 Ω 数量级。电抗成分为电容,约几个 pF。ECL 电路的输出阻抗与 TTL 电路、CMOS 电路不同。ECL 电路采用射极跟随器输出,在高电平(1 态)和低电平(0 态)时的输出阻抗比较接近(均为数 Ω),TTL 电路和 CMOS 电路的不同电平时的输出阻抗则有较大差别。

为了吸收反射,减少传输线不匹配和沿线负载的不良影响,普遍采用匹配终端的方法,常用的匹配终端的方法有以下 5 种。

(1) 串联电阻 适用于负载集中在线的终端的情况。电阻串接在驱动源附近,其阻值为负载传输线特性阻抗和驱动源内阻之差。

(2) 并联电阻 此方法应用广泛。将阻值等于负载传输线特性阻抗的电阻一端接在传输线终点上,电阻另一端接地或接电源 V_T 。在 ECL 电路中, $V_T = -2\text{ V}$; 在 TTL 电路中, $V_T = +3\text{ V}$ 或 $+5\text{ V}$ 。

(3) 分压电阻 适用于 TTL 电路。传输线终点上接有两个电阻,其中一个电阻另一端接 $+5\text{ V}$,另一个电阻的另一端接地。此方法本质上等效于并联电阻,常用于时钟信号线和总线上。

(4) 阻容网络 在 TTL 电路和 CMOS 电路中能很好地工作。此方法是将电阻和电容串联接在传输线终点与地之间,电容值在 200 pF ~ 600 pF 范围内。电容与电阻形成的时间常数(RC 值)必须大于负载传输线延迟的两倍。

(5) 二极管网络 常用于差分网络中,它能够将过冲限幅在 1 V 以内,功耗小。

5 种方法各有利弊,要结合实际条件通过实验合理选择。

为了减小反射,在布线设计中推荐沿线分布负载,避免 T 型分叉线,管座和过孔应分别增加 5 pF 和 2 pF 负载;要计入分布电容的计算;建议少用管座,减少过孔;在高性能的数字系统中要控制特性阻抗。减少反射的根本出路在于缩短传输线长度,提高集成电路的集成度,采用高密度组装技术。

时域反射仪(TDR)是观察和测试反射的仪器,它能显示传输线不连续的地点和程度。

串扰

串扰是高速信号在传输线上传输时,由于互感和电容的存在,在邻近传输线中感应的噪音信号。设有一个数字脉冲在两根相邻的特性阻抗为 Z_0 的平行传输线之一中传输,两线互相耦合的电容和互感为 C_m 和 L_m ,在驱动线上脉冲到达某一点 a 引起电容电流 i_c 在感应线中流动。它分成数值相等方向相反的两部分,且各向着感应线的两端流动。驱动脉冲前进的方向称为前向,其相反方向为后向。根据楞次定律,感应线上还感应出电感电流 i_l 阻止驱动电流的流动, i_l 总是流向后向。因此在 a 点有 $i_l + i_c$ 流向后向, $i_l - i_c$ 流向前向, i_l 和 i_c 都正比于驱动电流函数的导数。

在感应线的后向终点的端电压与驱动源极性相同,电流 $i_l + i_c$ 在负载上形成了后向串扰电压。在感应线的前向终点,由于 i_l 和 i_c 的方向相反,但大小不会相等,电流 $i_l - i_c$ 也会产生一个前向串扰电压。长线的后向串扰电压的幅度正比于驱动电压的幅度,宽度等于传输延迟的两倍,即 $2T_d$ 。前向串扰电压幅度正比于驱动信号对时间的导数和耦合长度,宽度等于驱动信号的上升时间。

在工程中,传输线两端的匹配终端情况和沿线负载不同,再加上传输线是非均匀介质,串扰会更加复杂。

在实际应用中减少串扰的方法是限制耦合长度和增加平行线间距离。在多层印制板中每两层信号线层间用一层地和电源层隔离,两层信号线互相垂

直。连接器的引线之间的串扰不可忽视,因而在高速应用中要用电缆中一部分引线两端接地,以便隔离引线间的串扰。带状电缆的信号线中要有一定比例的地线插在信号线之间,多条带状电缆应避免平行重叠放置。(陈鸿安)

gaoxingneng jisuan

高性能计算 (high performance computing)

用高速计算机系统解决复杂问题的计算方式。由于高性能计算通常使用并行计算技术,因此高性能计算体现为各种不同形式的并行计算。高性能计算可以解决原来单靠理论方法或者实验方法无法解决的问题,并且在传统的理论与实验方法之间架起了相互联系的桥梁。高性能计算在现代科学研究、国民经济和社会发展中的作用越来越重要。

高性能计算机的发展阶段与结构分类

高性能计算机发展的几个阶段和不同的体系结构密切相关。

高性能计算机发展的第一个阶段是 20 世纪 70 年代中期以 Cray 1 超级计算机为代表的向量计算机(参见向量计算),这种超级计算机主要由特殊的向量处理器和专门的操作系统构成。向量计算机后来进化为并行向量处理(PVP)高性能计算机。

第二个阶段是在 20 世纪 90 年代的对称式多处理(SMP)(参见共享存储)、高速缓冲存储器一致性非均匀存储器访问(CC-NUMA)(参见共享存储)以及大规模并行处理(MPP)(参见大规模并行处理)等结构。SMP 与 CC-NUMA 都是共享存储系统,它具有如下特点:①对称性,即任何一个处理器都可以访问本系统的任何内存单元和外部设备;②单一地址空间,即不存在局部与全局地址空间的问题,所有地址和数据都在一个相同的空间中;③存储器通信,和 I/O 通信相比,可以具有较高的通信效率。CC-NUMA 是分布式共享存储系统,它是 SMP 系统的扩展,既可以保持 SMP 结构的优点,又增强了扩展性。MPP 与 SMP 和 CC-NUMA 的最大不同点在于其物理内存的分布性,同时其处理器个数也可以比较多,MPP 高性能计算机的处理器可以成千上万。

第三个阶段是 20 世纪 90 年代中后期发展起来的集群式高性能计算机系统(参见集群式计算),它采用成品化的部件,其特点是性能价格比较高,同时研制周期比较短,有很好的扩展性。它一般是由工作站或者高档微机组组成计算结点,通过高速互联网

络连接起来,采用商品化的操作系统或者免费操作系统与软件工具。另有一种高档次的集群式高性能计算机系统,它是以 SMP 或者 CC-NUMA 等高性能计算系统为超结点而构成的系统,可以提供非常高的计算能力。

高性能编程模型

高性能计算需要高性能的并行编程模型,主要的并行编程模型有共享变量、数据并行和消息传递三种,另外还有面向对象、函数式与逻辑编程模型等。下面介绍主要的并行编程模型。

在共享变量模型中,多个并行成分可以对相同的共享变量进行操作,它具有如下特点:①单一地址空间,由于共享变量与数据在相同的地址空间中,因此不需要显式数据分配,各个进程或线程可通过访问公共存储器中的共享变量来进行通信;②多线程,这需要编程者设计不同的并行执行语句,来实现程序的并行执行;③异步,线程之间需要显式的同步语句,来保证各个线程有正确的执行顺序。这种模型的典型代表是 OpenMP 标准,此外还有 Pthreads 等。和消息传递编程模式相比,共享变量程序的移植性相对较差。

在数据并行模型中,不同处理器同时对不同的数据执行相同的操作,它具有如下特点:①全局名字空间,从概念上说,数据并行模型提供给各个不同处理器的是共享内存,它们具有相同的地址空间,因此不需要消息通信语句,不同处理器之间的通信是通过访问共享内存实现的;②单线程,编程者将数据并行程序看作是单个线程的执行,而不是多线程的执行,具有类似串行程序的控制流;③松同步,即数据并行程序的同步是语句级的,而不是指令级的。针对数据并行编程模式,有相应的并行语言,常用的是高性能 FORTRAN(HPF)。它通过引入标注语句,让编程者负责实现对数据的划分,同时引入一些常见和方便操作的并行结构,用来表达各种数据并行执行形式。这种方式的优点就是可以提高编程的级别,提高编程效率,缺点就是对于一些非典型数据并行问题,一般不易表达,而且性能也不高。

消息传递是另外一种重要的并行编程模型,它的特点是:①多进程,从程序员的角度看,消息传递并行程序表达的是多个进程的并行执行;②独立地址空间,即各个进程都有自己独立的编址空间,不同进程之间相互独立;③异步执行,各个进程之间的同步是通过显式的消息传递语句实现的,它们之间完全可以异步执行。消息传递编程模型的一种常用

标准是消息传递接口(MPI)。用MPI来编写并行程序,可以取得较高的通信效率,但是编程者必须负责完成不同处理器之间的通信操作,这样就增加了编程者的负担,影响了并行程序的开发效率。

高性能计算工具与环境

高性能计算必须有相应的工具和环境的支持。程序开发工具和环境有助于高效开发高性能的应用程序,包括程序编辑器、并行程序编译器、并行程序调试器、并行系统软件和支撑语言等。

(1) 并行编译器 一个并行编译器主要由3部分组成:流分析、程序优化和代码生成。流分析是确定源代码中数据和控制的相关性;优化是将代码转换成与之等效但更好的形式,以利于挖掘硬件潜力,最终达到全局优化的目的;代码生成涉及到从一种描述转换到另一种中间形式的描述。主要的并行编译器有两种,即自动并行编译器和人工辅助并行编译器。并行编译有向量并行、函数并行、指令并行等不同的并行层次。还可为并行计算机开发智能编译器,但难度很大,离实际需求相差甚远。

(2) 并行调试器 并行程序的调试和分析很困难。调试的目的是为了获得一个正确的并行程序。目前并行程序调试的技术和手段不成熟,其主要原因在于并行化的程序语句执行的次序是不确定的。这种不确定性意味着特殊的机器指令执行序列是不可重现的,因此难以跟踪观察。主要的调试手段有断点调试、事件分析和重放。

(3) 并行程序性能分析 并行程序的性能分析工具主要用来分析并行程序的性能,提供性能参数,指导程序的优化和改进系统的设计。性能分析工具一般分为静态和动态两种:前者采用模拟或分析方法获取源程序中的有关性能数据;后者采用测量的方法收集程序运行中的各种性能数据,即时或事后报告给用户。动态分析提供的数据比较准确,但灵活性较差;静态分析能够针对不同的程序和运行环境给出性能预测,但准确性有待提高。

发展趋势

半导体芯片是高性能计算机的核心器件,微电子技术的革新是50年来推动计算机技术发展的最根本的驱动力。然而,传统的以硅为基础的芯片制造技术的发展不是无限的,由于存在磁场效应、热效应、量子效应以及制作上的困难,因此需要开拓新的芯片制造技术。以新技术为基础的未来高性能计算包括分子计算、光计算(参见光计算机)、生物计算(参见生物计算)和量子计算(参见量子计算)等。

这些技术离实用还有相当距离,但是由于新技术具有诱人的潜力,因此引起人们的广泛注意。

参考文献

1. Kai Hwang, Zhiwei Xu. Scalable Parallel Computing Technology, Architecture, Programming. McGraw-Hill, 1998
2. 陈国良. 并行计算——结构·算法·编程. 北京: 高等教育出版社, 1999 (陈渝 都志辉)

Gedeer peishu

哥德尔配数 (Gödel numbering) 使用素数幂的乘积来表示自然数的任意的有限序列 x_0, x_1, \dots, x_n 的所有方案。表示本身称为序列 x_0, x_1, \dots, x_n 的哥德尔数, 通常记为 $\langle x_0, x_1, \dots, x_n \rangle$ 。虽然每种方案都有自己的长处与不足, 但是, 经常使用的是所谓标准方案。在标准方案中, 序列 x_0, x_1, \dots, x_n 用自然数 $P_0^{x_0} P_1^{x_1} \dots P_n^{x_n}$ 来表示, 其中 P_i 表示第 i 个素数。按标准方案, 许多不同的序列可能具有相同的哥德尔数。定义二元函数 E , 使得:

$$E(i, z) = \mu^i x [\text{div}(P_n(i)^{x+1}, z) = 0]$$

其中: 二元函数 div 定义为: 若 $x > 0, y > 0$ 且 x 整除 y , 则 $\text{div}(x, y) = 1$; 否则 $\text{div}(x, y) = 0$ 。 $P_n(i)$ 是第 i 个素数。若 z 是序列 x_0, x_1, \dots, x_n 的哥德尔数, 则对每个 $0 \leq i \leq n$ 有: $E(i, z) = x_i$; 对每个 $i > n$ 有: $E(i, z) = 0$ 。函数 E 称为提取函数, 值 $E(0, z), E(1, z), \dots, E(i, z)$ 称为 z 的分量。定义一元函数 Lh , 使得:

$$Lh(z) = \mu^i n \left[\prod_{i=0}^n P_n(i)^{E(i, z)} = z \right]$$

则 $Lh(z)$ 是整除 z 的最大素数的下标。函数 Lh 称为长度函数, $Lh(z)$ 的值称为 z 的长度, 常记为 $|z|$ 。

为了对不同的序列提供不同的表示, 经常使用标准方案的以下两种变形: 在第一种变形中, 序列 x_0, x_1, \dots, x_n 用自然数 $P_0^{x_0+1} P_1^{x_1+1} \dots P_n^{x_n+1}$ 来表示。按这种方案, 许多自然数根本不表示任何序列。在第二种变形中, 序列 x_1, x_2, \dots, x_n 用自然数 $P_0^{x_1} P_1^{x_2} P_2^{x_3} \dots P_n^{x_{n+1}}$ 来表示。按这种方案, 并非每个自然数都表示某个序列。除此之外, 还有许多其他的变形。

哥德尔配数为长度不等的所有有限序列规定了一个顺序。许多重要定理的证明都用到哥德尔配数。

参考文献

- Hennie F. Introduction to Computability. Massachusetts: Addison-wesley, 1977 (殷建平)

Gedeeer wanquanxing dingli

哥德尔完全性定理 (Gödel's completeness theorem)

K. Gödel 于 1930 年证明的谓词演算 (参见一阶逻辑) 具有完全性的定理。谓词演算具有可靠性, 即它的每个定理都是逻辑有效式。哥德尔完全性定理说: 谓词演算具有完全性, 即每个逻辑有效式都是谓词演算的定理。

谓词演算是没有非逻辑公理的一阶理论, 可以将哥德尔完全性定理推广到一般的一阶理论。若公式 A 在一阶理论 T 的每个模型中有效, 则称 A 在 T 中有效, 记为 $T \models A$ 。 A 是 T 的定理记为 $T \vdash A$ 。可以证明, 对于一阶理论 T 的公式 A , $T \models A$ 当且仅当 $T \vdash A$ 。人们也把它称为现代哥德尔完全性定理。这个定理成立的根本原因在于: 逻辑有效式的集合是递归可枚举集和一阶逻辑具有紧致性 (参见模型论)。

如果有一个公式 A 使得 A 和 $\neg A$ 都是一阶理论 T 的定理, 则称 T 是不协调的, 否则称 T 是协调的。哥德尔完全性定理的另一等价形式是: 一阶理论 T 协调的充分必要条件是 T 有模型。哥德尔完全性定理说明, 一阶逻辑的形式系统完全正确地反映了直观的逻辑推理。它把一阶逻辑的语法性质 (公式的可推出性) 和语义性质 (公式的有效性) 联系起来。表明二者是一致的。这为用语义方法研究一阶理论的语法性质, 以及用语法方法研究一阶理论的语义性质提供了理论根据, 为模型论的诞生准备了条件。由哥德尔完全性定理可以直接导出一阶逻辑的紧致性。由谓词演算的定理集是递归可枚举集和哥德尔完全性定理可以得出, 逻辑有效式的集合是递归可枚举集。判断一公式的逻辑有效性是人工智能中需要经常面对的问题, 判断公式逻辑有效性算法理论基础的埃尔布朗定理, 正是一阶逻辑紧致性定理的直接推论。

参考文献

1. 王宪钧. 数理逻辑引论. 北京: 北京大学出版社, 1982
2. Kleene S C 著. 元数学导论. 莫绍揆译. 北京: 科学出版社, 1984 (何自强)

ge

格 (lattice) 一种特殊的偏序集。

设 $\langle L, \leq \rangle$ 为一个偏序集。若对任意 $a, b \in L$, L 的子集合 $\{a, b\}$ 都有最小上界和最大下界, 则称 L 是一个格, 并把 $\{a, b\}$ 的最小上界和最大下界分别

记为 $a \cup b$ 和 $a \cap b$, 分别称为 a 与 b 的并和 a 与 b 的交。若把 \cup 和 \cap 看作 L 上的两个二元运算, 则它们满足以下运算定律:

交换律: 对任意 $a, b \in L$, 皆有 $a \cup b = b \cup a$ 和 $a \cap b = b \cap a$;

结合律: 对任意 $a, b, c \in L$, 皆有 $(a \cup b) \cup c = a \cup (b \cup c)$ 和 $(a \cap b) \cap c = a \cap (b \cap c)$;

吸收律: 对任意 $a, b \in L$, 皆有 $a \cup (a \cap b) = a$ 和 $a \cap (a \cup b) = a$ 。

反过来, 若在集合 L 上定义了满足上述交换律、结合律和吸收律的两个二元运算 \cup 和 \cap , 则可在 L 上引出一个偏序: 对任意 $a, b \in L$, $a \leq b$ 当且仅当 $a \cup b = b$ (当且仅当 $a \cap b = a$)。可以证明, $\langle L, \leq \rangle$ 是一个偏序集, 且对任意 $a, b \in L$, $\{a, b\}$ 在 L 中有最小上界和最大下界, 即 $\langle L, \leq \rangle$ 是一个格。这样一来, 既可以把格看作一种特殊的偏序集, 又可以把格看作具有满足交换律、结合律和吸收律的两个二元运算的代数结构。

如果 L' 是格 L 的非空子集, 对任意 $a, b \in L'$ 皆有 $a \cup b \in L'$ 和 $a \cap b \in L'$, 则称 L' 为 L 的一个子格。设 $L = \{1, 2, 3, 6, 12\}$ 且 $L' = \{1, 2, 3, 12\}$, 并定义半序 \leq 为: 对任意二正整数 a 和 b , $a \leq b$ 当且仅当 b 能被 a 整除。这时 L 和 L' 显然都是格。但 L' 不是 L 的子格, 虽然有 $L' \subseteq L$ 。

一个包含格的元素和符号 $=, \leq, \geq, \cup, \cap$ 的命题的对偶命题, 是指用 \geq, \leq, \cap, \cup 分别代替该命题中的 \leq, \geq, \cup, \cap 而获得的命题。格的对偶原理是: 若命题 P 在任意格中都成立, 则 P 的对偶命题也在任意格中成立。

如果格 L 中有元素 0 和 1 , 且对每个 $a \in L$ 皆有 $0 \leq a \leq 1$, 则称 L 为有界格, 0 和 1 分别称为 L 的上界和下界。

设 A 为任意集合, 则 A 的幂集 $\mathcal{P}(A)$ 关于集合的并运算和交运算构成一个有界格, 其中 $0 = \emptyset$ 且 $1 = A$ 。

设 L 为有界格且 $a \in L$ 。若 $b \in L$ 使 $a \cup b = 1$ 且 $a \cap b = 0$, 则称 b 为 a 的一个补元, 这时也称 a 有补元。当 a 为 b 的补元时, a 亦是 b 的补元。若格 L 的每个元素都有补元, 则称 L 为有补格。格中一个元素的补元不一定惟一。若格 L 中每个元素均有惟一的补元, 则称 L 为惟一有补格。

如果对格 L 中任三元素 a, b 和 c 皆有 $a \cap (b \cup c) = (a \cap b) \cup (a \cap c)$ (此时自然也有 $a \cup (b \cap c) = (a \cup b) \cap (a \cup c)$), 则称 L 为分配格。当分配

格中的一个元素有补元时,其补元必是惟一的。

如果 L 为有穷集合,则称格 L 为有限格,否则称格 L 为无限格。

如果格 L 的每个子集合都有上确界和下确界,则称 L 为完备格。有限格显然是完备格。

有补分配格称为布尔代数。

设 L 为格。若对任意 $a, b, c \in L$, 当 $a \leq c$ 时皆有 $a \cup (b \cap c) = (a \cup b) \cap c$, 则称 L 为一个模格或戴德金格。分配格显然是模格。

参考文献

1. Bwrris S and Sanppanavar H. A Course in Universal Algebra. New York: Springer-Verlag, 1981

2. 王兵山, 周贤林, 王长英, 何自强编. 离散数学. 长沙: 国防科技大学出版社, 1985 (王水汀)

Geleibeiqi fanshi

格雷贝奇范式 (Greibach normal form)

一种由 S. A. Greibach 提出的上下文无关文法的规范化形式。它对上下文无关文法的生成式形式加以某种限制。S. A. Greibach 证明了任意的上下文无关文法都存在一个等价的格雷贝奇范式。

若上下文无关文法 $G = (\Sigma, V, S, P)$ 中的生成式均为 $A \rightarrow a\alpha$ 的形式, $a \in \Sigma, A \in V, \alpha \in V^*$, 则称 G 为格雷贝奇范式, 缩写为 GNF。格雷贝奇在 1965 年证明了任何不含 ϵ 的上下文无关语言都可由一个格雷贝奇范式产生, 这就是著名的格雷贝奇范式定理。M. C. Paull 和 D. J. Rosenkrantz 等人分别给出了将任意的上下文无关文法转换成等价的格雷贝奇范式的算法。

虽然对任意的上下文无关文法, 都可进行简化, 但根据格雷贝奇范式定理还可构造一个等价的格雷贝奇范式。这种范式不仅使相应的上下文无关文法的生成式异常简单而且规范。这对研究相应的上下文无关语言的结构很有意义, 例如, 由 GNF 产生的每个串, 其长度与产生它的推导步数相等。

参考文献

Hproft J E, Ullman J D. Introduction to Automata Theory, Languages, and Computation. Addison-wesley Publishing Company, 1979 (苏锦祥)

ge yufa

格语法 (case grammar) 1968 年由 Charles Fillmore 在其论文《“格”辨》中创立的一种面向语义

的语法理论。

C. Fillmore 认为一个句子 (S) 由情态 (M) 和命题 (P) 两部分组成, 其中命题以述语动词 (V) 为中心, 句中各名词性成分所具有的语义功能可根据它们与该动词的语义关系确认为不同的格 (C_i), $i = 1, 2, \dots, n$ 。所以, 格语法可形式化地表示为:

$$S \rightarrow M + P$$

$$P \rightarrow V + C_1 + C_2 + \dots + C_n$$

在传统语言学中, “格”是指某些屈折语中名词和代名词的形态变化, 如主格、宾格、属格等表层格。C. Fillmore 所说的格是句子深层的语义格, 最初他根据英语提出的六种格是: 施事、受事、与格、使役、工具和处所。为避免术语上的混淆, 近来国际学术界将述语动词的格关系起名为论旨属性, 并把相应的深层格称作论旨角色。

格语法的一条基本原理是: 尽管表层句式不同, 只要它们包含的述语动词具有相同的义项和相同的格关系, 那么它们的句意表达式就应当是惟一的。比如, 动词“撕”的基本句式是:

我撕了那封信。

其句意可用格框架表为:

(撕:〈施事〉= 我, 〈受事〉= 那封信,

〈时态〉= 过去)

根据言谈者的不同意图, 上述例句可变换成如下不同的表层句式:

我把那封信撕了。

那封信被我撕了。

那封信我撕了。

但它们的句意表达式的命题 (P) 部分保持不变, 言谈焦点 (或语用) 的差异仅反映在表达式的情态 (M) 部分。句子命题的这种归划一化表示使计算机有能力回答诸如“是谁撕的?”, “撕了什么?” 等仅凭借句子的句法结构难于回答的问题。

为了在自然语言理解中用格语法来分析或生成句子, 不仅需要在电子词典中对每个动词词条的不同义项分别给出其格框架的描写, 而且要对其中的每个格角色给出语义上的选择性限制。例如, “撕”的施事是〈人〉或〈动物〉, 受事是〈片状脆弱物〉等。这意味着我们还需要建立一部概念词典 (或义类词典), 以便计算机能据此判断述语动词周围的名词性成分是否能担当特定的格角色。

鲁川、林杏光合作编著的《动词大词典》根据汉语的特点定义了 22 个格, 构成如图 1 所示的格系统:

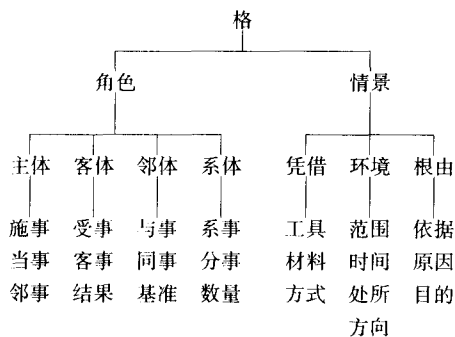


图1 格系统

这部词典对现代汉语的1000多个常用动词的2000多个义项的格框架进行详细的描写,并给出了相应的基本句式、扩展式及变换式。

从计算机对自然语言的处理来看,述语动词和形容词是句子句法结构和语义解释的中心,采用格语法比较容易实现从句法分析到句意表达的过渡。因此,格语法已被广泛应用于各国的自然语言处理系统中,并成为机器翻译中中间语言的基础。这也说明,格语理论不依赖于特定语种,而具有语言上的普遍性。

参考文献

1. Fillmore C J 著.“格”辨.语言学译丛.胡明扬译.1980,2;1~117
2. 林杏光审定,鲁川主编.动词大词典.北京:中国物资出版社,1994 (黄昌宁)

geren shuzi zhuli

个人数字助理 (personal digital assistant, PDA) 可以装在衣服口袋里用来帮助人们记录处理商务和日常活动数据的计算机。个人数字助理外形尺寸一般在8 cm×12 cm×1 cm左右或者更小,质量一般在100~350 g或者更轻。个人数字助理的基本配置包括微处理器、存储器、显示器、键盘、接口、电源、系统软件及应用软件等。

个人数字助理的种类很多,分类方法也多种多样。一般可以从性能和用途进行分类。个人数字助理性能的外特性主要包括显示器的分辨率,使用电池的型号与电源消耗的快慢,数据交换能力,通信能力等;性能的内特性(系统特性)包括系统的软硬件配置和软件开放性(兼容性)。系统配置包括微处理器的型号,内存的大小,各种接口等。个人数字助理的核心技术包括:缩小尺寸的硬件设计、省电的

电源管理、嵌入式软件、数据的输入与输出等。个人数字助理产品一般包括:电子词典、电子记事本、手持计算机等。

电子辞典类的个人数字助理主要包括计算器、英汉汉英词典、电话号码簿、游戏软件等。此类设备的优点是价格便宜,体积小,重量轻,耗电量低;缺点是功能弱,可允许用户记录的数据种类和数据长度有限。为了控制成本和耗电量,该类产品中使用的微处理器芯片一般是8位或4位的微处理器,显示器使用黑白和低分辨率的液晶显示器,电源使用7号电池或者纽扣电池,键盘使用电子计算器类似的键盘。此类产品的功能和软件一般是固定的,不能升级。

电子记事本类的个人数字助理的产生起源于人们希望用计算机代替笔记本,因此使用笔输入代替键盘成为此类设备的关键要求。此类设备的优点是体积小,重量轻,应用软件较丰富,耗电量较低;缺点是功能较弱,系统开放性不好,与台式计算机软件无法兼容,网络与通信能力较差。该类设备一般使用16位的微处理器芯片,显示器使用黑白160×160或320×240等较高分辨率的液晶显示器,电源使用5号或7号AA电池,输入使用触摸屏代替键盘。其软件大都具备基本的个人信息管理功能,主要包括电话本、备忘录、日程管理、个人理财、万年历、汇率转换、计算器、电子词典、口语练习、辅助写作、电子游戏等。此类产品一般具有和计算机之间的数据交换、数据同步和应用程序交换的能力,软件一般可以升级。比较高级的电子记事本还具有收发电子邮件和传真等的功能。

手持计算机的尺寸与电子记事本相近或略大一些。此类设备的核心设计思想是软件可以与笔记本计算机和台式计算机兼容,所以需要考虑使用通用操作系统或经过剪裁的通用操作系统。此类设备的特点是开放性好,软件丰富,功能强。缺点是耗电量较高,成本高。该类设备一般使用32位的微处理器芯片,显示器使用彩色高分辨率的液晶显示器,电源使用5号AA电池或专用电池,输入使用触摸屏代替键盘。这种计算机所支持的应用软件种类非常丰富,包括办公套件、游戏、网络应用、视频音频播放软件等。

(高文)

geren tong xin wang

个人通信网 (personal communication network, PCN) 提供个人通信业务(PCS)的任何网

络。**个人通信业务 (PCS)**是指利用对通信网络透明的个人电信号码,经过传输网和智能网的接入、传输、交换等处理可实现任何用户在任何时刻、任何地点能够同其他任何用户实现自己预定的任何电信业务组合服务的通信。个人通信的能力仅受通信终端网络能力及网络经营者所施加限定的约束。

个人通信的主要特征是:通信到用户个人而不是设备终端;以个人电信号码为惟一专用的通信识别码;以个人用户为通信计费单位,而不是根据终端或用户线来计算;有极大的用户容量;用户能随时入网,无论室内室外,无论有线无线;通信发射功率极低,频带能重复利用;设备功耗小,体积小,便于携带;提供各种信息通信服务,包括信息安全通信;有效的用户跟踪管理和网络管理等。

目前,个人通信的研究发展重点主要集中在三类个人通信网上。

(1) 户内、户外和高大建筑内步行者的个人通信 适合于低移动速度,高用户密度的使用环境。在无绳电话基础上发展起来的个人通信和室内无线局域网属于这一类。

(2) 高速移动或远区用户的个人通信 适合于相当高的移动速度应用环境,高用户密度与低用户密度均能支持。**全球移动通信系统 (GSM)**属于这一类。

(3) 卫星移动通信用户的个人通信 以卫星作为基站和中继转发器,是实现广域、国际、洲际联结的,真正做到在世界上任何地方、任何时刻,对任何人以任何形式进行通信的全球个人通信网。适合于边远地区、落后地区、山区及野外作业等环境。

个人通信研究发展的重点是无线个人通信,其涉及的关键技术主要是无线接口技术、用户与通信网的多址技术、分布式智能数据库等。

个人通信与移动通信两者的不同主要表现在:
①移动通信一般仅限于一定的区域,而个人通信是全球性质的;
②移动通信一般只具有终端移动性,个人通信则具有个人移动性;
③移动通信一般只能提供窄带业务,且以话音业务为主,而个人通信却可以提供综合业务,并具有宽带传输能力;
④移动通信可以以一个移动通信网的形式独立存在,而个人通信则是包括了有线通信网和无线通信网;
⑤个人通信网有非常大的通信容量和智能化管理以满足每个人的通信要求;
⑥个人通信以个人电信号码为惟一专用的通信识别码,而移动通信中则无此要求;
⑦移动通信一般是有缝的,很难在各个移动通信系

统之间实现漫游,而个人通信能在全球范围内为任何人提供无缝的通信服务。

参考文献

1. 谈振辉. 个人通信的进展和研究. 数字通信, 1995(1)
2. 朱近康. 个人通信网和个人通信服务. 电信科学, 1994, 10(8)
3. 张更新. 移动通信与个人通信. 电信科学, 1996, 12(3) (史美林 英春)

genzongqiu

跟踪球 (track ball) 一种用于控制屏幕上光标运动的输入设备。它由一个露出一半可以自由转动的圆球及其 x 方向和 y 方向的轴角编码器组成。实际上可看成一个底朝天的鼠标器。跟踪球和鼠标器在功能上是相同的,都是把球的运动方向和距离变成脉冲信号送入计算机,控制屏幕上光标的运动。它们之间的不同点在于鼠标器的操纵球是向下露出一半,鼠标器移动时靠摩擦带动圆球运动;而跟踪球的操纵球是向上露出,由使用者用手指拨动圆球运动。另外,跟踪球的检测传感器除了和鼠标器一样采用光学编码器等外,还可以采用压力传感器。

跟踪球占用的面积较小,常附设在便携式电子计算机的键盘旁,可以很方便地用一只手操作,因此使用越来越多。 (林兼)

gongcheng shujuk

工程数据库 (engineering database) 为工程应用的特殊需要而设计的数据库。除了数据库的一般功能外(参见**数据库系统**),工程数据库要解决的两个主要问题是:复杂的工程数据的表达、处理以及提供工程应用所需的功能。如,大量复杂数据的高效存储和访问功能、长事务管理功能、版本管理功能等。

工程数据库是随着数据库应用的日益广泛深入而发展起来的。早期的系统不强调数据和程序的分离,一般以文件作为持久数据的存储手段。后来出现了**数据库管理系统 (DBMS)**,数据管理的手段大大加强了。20 世纪 70 年代中,数据库开始应用于工程领域,典型的如**计算机辅助设计 (CAD)**、**计算机集成制造系统 (CIMS)**等。在这些领域中,由于关系数据模型及其 DBMS 难以满足需求,专为工程领域而设计的工程数据库应运而生。目前的工程数据

库主要有两种实现方式:一是在关系 DBMS 的基础上加以扩充或改进;二是开发支持新数据模型的数据库管理系统,如面向对象数据库管理系统。目前,工程数据库已广泛应用于各主要工程领域。

工程数据库具有以下特殊要求:

(1) 数据模型 在工程领域,需支持复杂对象的表达和处理以及丰富的数据类型。一般的数据模型为层次型、网状型、关系型,均由于其较弱的建模能力而不能完全适应需要。正在发展中的面向对象数据模型(参见面向对象数据库)一般被认为是工程数据库数据模型的合理选择。但在目前实际应用中,由于关系 DBMS 有成熟的商品供应和维护服务的支持,仍有不少领域用关系数据模型来建立工程数据库。

(2) 存储管理 存储管理是工程数据库的基础,为其他模块提供底层的基本支持。其设计和实现关系到整个系统的运行效率和可靠性。在存储管理中,经常采用缓冲技术,以提高存取效率。缓冲技术中最关键的是调度算法和内外存一致性的保证。对调度算法的要求是一要可靠,二要高效。在内外存一致性方面,要求系统不论何时都不会因任何原因而产生不可恢复的数据不一致。在这方面,存储管理和事务管理是紧密联系在一起的。

(3) 事务管理 事务管理的主要功能是并发控制和故障恢复。事务是数据库操作的基本序列。事务管理的基本要求是:数据库的操作或是全做或是全不做,以保证在事务开始和结束时数据库的一致性。

和商用事务相比,工程事务具有长期性、协作性和试探性。为支持这些特点,必须摒弃商用事务处理中简单的等待或回卷。既不能让用户因合理的申请而长时间等待,也不能简单地对失败的操作进行回卷,而是应建立一套新的机制以支持工程事务的长期性和试探性特点。

事务的并发功能是指不同的事务在执行时间上有所重叠。其中关键的技术是解决并发所引起的冲突。一般工程数据库都以锁定来解决冲突。工程长事务所持有的锁可以是持久锁。

(4) 版本管理 工程设计具有试探性、往复性、共享性,经常在同一对象的不同版本间进行切换。因此,版本管理是工程数据库不可缺少的功能。版本反映对象的演变过程。版本的表达方式必须根据实际应用背景的要求选择,并提供版本生成、删除和切换的功能以及复合对象的版本管理。

(5) 查询处理 查询功能要求处理能力强、效率高。在工程数据库中,为适应实际需要,存在两种对数据库进行查询的方式。

一种是联想查询,也就是关系数据库所采用的查询方式。即根据用户提供的条件,在数据库的当前环境中查询检索符合条件所有对象。

另一种是导航式查询。由于工程数据库具有数据量大且数据间关系复杂的特点,而联想查询不仅速度很慢,而且对复杂的对象往往很难构成合适的查询条件。因此,除了联想查询外,在工程数据库中还提供导航式查询。它主要针对复合对象的查询。利用复合对象中对象的层次构成查询路径。查询的路径可以是很复杂的。查询过程犹如领航员导航一样。

实际的工程数据库查询处理是两者在给定应用条件下的合理折衷。

随着工程数据库应用的不断深入和发展,工程数据库的功能也将日益增强,并将与应用领域有更密切的联系。在数据模型方面,将进一步发展表达能力更强的模型。在存储方面,工程数据的存取效率将是今后研究的课题。在查询方面,将实现两种查询方式的紧密结合并提高查询效率。

参考文献

Cattell R G G. Object Data management: object-oriented and extended relational database systems. Addison-wesley, 1991

(寿宇澄 孙建伶)

gongjuxiang

工具箱 (toolkit) 用于软件系统开发的一组相关软件工具。

软件开发工具的研制始于 20 世纪 70 年代。早期的软件工具均是单独使用的,完成各自独立的工作。70 年代中期,出现了工具箱概念,其基本思想是将一组相关的软件工具放在一起以便于用户使用,提供对用户开发工作的整体和系统的支持。

一个典型的工具箱实例是 OSF/Motif,它是一个具有可构造特性的交互式用户界面工具箱,用户可以使用 Motif 方便地生成自己所需的图形界面。

工具箱中的工具间缺少紧密的有机联系,工具箱本身也不像软件开发环境那样具有较强的指导用户如何去使用它的能力,用户本身的能力及其对工具箱的了解熟练程度决定了用户能否对工具箱的功能充分发掘并很好使用。

工具箱的出现是软件开发环境的萌芽,从工具

箱发展到开发环境只经历了很短时间。由于工具箱能为用户提供较大的自由度和灵活性,因此,其存在是必要的,将来仍会继续存在。但是,工具箱的用户友善性将会不断改善。(梅宏 邵维忠)

gongye kongzhi jisuanji

工业控制计算机 (industrial control computer)

按常见工业现场环境条件设计,适用于工业过程实时监视、控制用的计算机。这是一种“流行”的或狭义的定义,尤其对于它的简称“工控机”,更适合于这一定义。它的广义定义,可以指用于工业控制的计算机,这包括过程控制计算机,和生产线控制、机械加工控制的计算机,它们可以是各类专用机和通用机。大至具有多级通信网络的集散型控制系统,小至带有单片机或微处理器的仪表型控制器。其主要特点如下:

(1) 可靠性高 要求在工业现场的恶劣环境条件(如高温、低温、高湿度、多粉尘、含腐蚀性气体、强电磁场干扰等)下,仍能可靠地连续运行,具有足够长的平均故障间隔时间(MTBF)。提高可靠性的措施主要有:①选用高质量的元器件并降级使用;②抗干扰的电路设计和可靠的线路板设计;③使用优质的工业电源;④采用监视定时器的自启动电路;⑤抗恶劣环境的系统结构设计;⑥具有抗恶劣环境并适合工艺操作的键盘或操作台;⑦具有后备设计,有的进一步采用容错及冗余设计等。

(2) 易维护性 系统结构上便于故障诊断和维修。新一代工业控制计算机具有在线维护功能,能按自诊断结果自动切断故障部分,可将故障模块或模块在线带电插、拔更换。

(3) 实时性强 有良好实时性的数据处理及通信能力。

(4) 易于扩展 适合工业现场较易变更控制方案、扩充控制回路数和功能的要求。

常见的工业控制计算机有以下几种类型:

(1) 专用工业控制计算机 以微处理器、单片机、单板机等为基础,它们往往以服务对象作为名称的修饰词,例如工业锅炉控制计算机等。这一类型的系统结构简单,价廉,可靠性可满足设计要求,但开发时针对性强,扩展性和通用性差。一般用于小型简单的过程监控系统。

(2) 扩展的工业控制计算机 以可编程控制器为基础,目前主要用于各种机械加工过程的控制,随着可编程控制器中模拟量 I/O 功能的引入,也能用

于其他的过程控制。其可靠性、扩展性和软件开发的方便性一般优于第(1)类。

(3) 模块化工业控制微型计算机 具有模块化结构,可按需要将一组具有特定功能的组件即模板或模块,用工业标准总线加以连接,构成系统。由于组件的模块化和标准化,这种系统的整机可靠性很高,系统设计灵活,易于扩展和维护。其中基于 PC 总线发展起来的工业控制微型计算机,具有极为丰富的软件支持,应用软件开发周期可大为缩短。

集散型控制系统中的现场控制站实际上也是一种工业控制计算机,但它一般没有人机接口,人机对话通过操作站进行。

参考文献

王常力. 工业控制计算机系统设计与应用. 北京:电子工业出版社,1993 (廖闻彬)

gongzuoqu zixitong

工作区子系统(work area subsystem) 结构化布线系统中将用户的终端设备连接到布线系统的子系统。工作区子系统包括各种不同型号的信息插座、适配器、连接跳线等将终端设备连到插座上所需的各种配件,如图1所示。

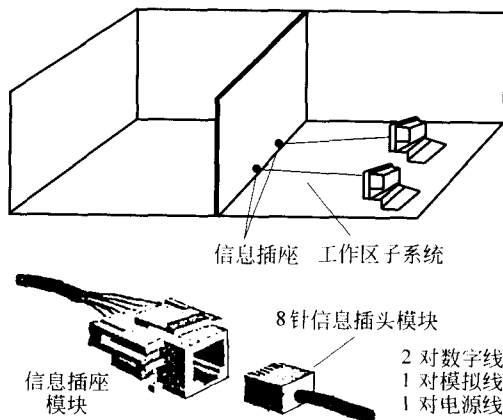


图1 工作区子系统

将一个独立的需要设置终端设备的区域划分为一个工作区。一个工作区的服务面积一般为 5 ~ 10m²。每个工作区设置一个电话插座和一个计算机插座。它们可以是专用插座或是信息插座。

信息插座是终端设备与水平子系统连接的接口。8 针模块化信息插座是为所有的结构化布线系统推荐的标准 I/O 插座。信息插座的选择由下列因

素决定:

(1) 用户通信质量及速率要求 可以根据用户不同的通信质量及速率要求选择三类、五类甚至是超五类插座。

(2) 插座安装环境要求 可以根据不同的安装环境选择桌面型、墙上型或是地面型插座,还可根据不同环境选择不同颜色插座。

插座的安装位置应考虑以下几个因素:

①尽量靠近使用者;②信息插座的安装要兼顾电源位置,在不造成相互干扰的情况下,一般情况下二者要尽可能近;③除了特殊情况,根据经验和有关电器安装规范,信息插座的安装距离地面的高度一般为 30 ~ 50cm。

信息插座的数量一般由使用者的数量所决定。如果使用者的数量不能确定,根据经验在办公环境下一般可考虑 9 平方米设置一个工作区,安装一对信息插座(一台电话、一台计算机)。但这仅是一个参考,在具体设计和施工过程中,设计单位和用户单位应根据具体情况灵活掌握。

参考文献

胡道元主编. 网络设计师教程. 北京: 清华大学出版社, 2001

(王晓东)

gongzuozhan

工作站 (workstation) 以个人计算环境和分布式网络计算环境为基础,其性能高于微型计算机的一种多功能计算机。个人计算环境是指为个人使用计算机创造一个尽可能易学易用的工作环境,为面向特定应用领域的人员提供一个具有友好人机界面的高效率工作平台。分布式网络计算环境是指工作站在进行信息处理的过程中,可以通过网络与其他工作站或计算机互通信息和共享资源。工作站的多功能是指它的高速运算功能,适应多媒体应用的功能和知识处理功能。高速运算包括中央处理器的高速定点、浮点运算以及高速图形和图像处理。多媒体应用是指工作站不仅能用于数值与文本数据处理,而且还能处理图形、图像、语音和声音。知识处理功能是指工作站能用于人工智能,如专家系统和基于知识的推理等。

构成工作站的硬件有主机、显示器和输入输出设备。按 20 世纪 90 年代中期的技术水平,主机包括:运算速度在几十 MIPS 到几百 MIPS 的中央处理器;主存储器容量至少是 8 MB 到 16 MB;有一级甚至二级高速缓冲存储器;磁盘容量从几 GB 到几十 GB

以上;机内设有各种总线接口、串行和并行接口、局域网和 FDDI 网接口、音频和视频标准接口等。显示器除了高分辨率的彩色显示器外,还包括显示控制器、帧缓冲存储器和图形处理器等,用以实现高速图形处理、三维动态显示以及实时仿真功能。输入输出设备主要有键盘、鼠标器和话筒等。此外,还可以接入磁盘、磁带、CD-ROM 以及图形、图像输入输出用的外围设备,如数字化仪、彩色扫描仪、绘图仪、摄录设备以及其他音频视频设备等。工作站的软件主要有:开放式操作系统(如 UNIX)和窗口系统;C, C++, FORTRAN 等主要语言的编译器;基本的工具软件和支撑软件有数据库、图形和图像处理软件以及网络软件等;另外,还有相应的应用软件。

发展简史

工作站的发展大致可划分为 4 个阶段。

第一阶段为 1973 年至 1979 年,是工作站实用化的研究开发阶段,典型产品有美国 Xerox 公司的 Alto 和 MIT 的 CADR 工作站。这个时期的工作站一般为:中央处理器采用 16 位微处理器,运算速度为 0.2 MIPS 左右;主存储器容量为 128 kB ~ 1 MB;显示器为单色,分辨率为 800 × 600 左右;网络为 3 Mb/s 以下的低速局域网;操作系统通常用 UNIX。

第二阶段为 1980 年至 1984 年,是工作站开始商品化并推广普及的阶段,典型的产品有 Apollo 公司的 DOMAIN, Xerox 公司的 STAR 和 Sun Microsystems 公司的 SUN-1。这个时期的工作站一般采用 16 位微处理器(MC 68000, MC 68010 等),运算速度为 0.2 ~ 0.5 MIPS;主存储器容量为 1 ~ 4 MB;显示器为单色,分辨率为 1 024 × 800;采用专用图形处理器;网络为 1 ~ 10 Mb/s 的中速局域网;操作系统为 UNIX 或类 UNIX。

第三阶段为 1985 年至 1986 年,是工作站迅速发展时期,典型的产品有 HP 公司的 HP 9000/320, Sun Microsystems 公司的 SUN-3 以及 SONY 公司的 NEWS。这个时期的工作站一般采用 32 位 CISC 微处理器,(如 MC 68020, MC 68030, Intel 80386 等),运算速度达 1 ~ 7 MIPS;主存储器容量为 4 ~ 32 MB;显示器为彩色(可 256 色),分辨率为 1 024 × 1 024, 64 种灰度等级(单色时),采用二维专用高速图形处理器;网络为 4 ~ 16 Mb/s 的中速局域网;操作系统为具有网络功能的 UNIX。

第四阶段为 1987 年以后,工作站采用 32 位甚至 64 位 RISC 微处理器,不仅处理速度快,而且具有强大的图形、窗口功能和界面,并向多处理机、开放

系统和分布式处理系统发展。典型的产品有 SUN 公司的 SPARC 系列和 DEC 公司的 Alpha AXP 工作站系列,以及 SGI, HP 等公司的工作站系列。这个阶段的工作站一般采用 32 位或者 64 位 RISC 微处理器,运算速度达 10 ~ 100 MIPS 以上;主存储器容量达 16 ~ 256 MB;显示器为彩色(可 2^{24} 种色),分辨率为 1280×1024 , 256 种灰度等级(单色时),采用三维专用高速图形处理器;网络为 4 ~ 16 Mb/s 的中速局域网或者 100 ~ 400 Mb/s 的高速局域网;操作系统为可支持多处理机并行处理的 UNIX SYSTEM V 或 OSF/1。

分类与应用

工作站的分类方法很多,按照工作站的用途可分为通用工作站和专用工作站。通用工作站没有特定的使用目的,可以在以程序开发为主的多种用途中使用。通常根据用途,在通用工作站上配置相应的硬件和软件以适应特殊用途。在客户-服务器环境中,通用工作站常作为客户机使用。专用工作站是为特定用途开发的,由相应用途的硬件和软件构成,可分为办公工作站、工程工作站和人工智能工作站等。办公工作站是为了高效地进行办公业务,如文件和图形的制作、编辑、打印、处理、检索、维护,电子邮件和日程管理等;工程工作站是以开发、研究为主要用途而设计的,大多具有高速运算能力和强化了了的图形功能,是计算机辅助设计、制造、测试、排版、印刷等领域用得最多的工作站。人工智能工作站用于智能应用的研究开发,可以高效地运行 LISP, PROLOG 等人工智能语言。后来,这种专用工作站已被通用工作站所取代。按照工作站的体系结构可以分为单处理机和多处理机工作站;按所用处理器的体系结构可分为复杂指令集计算机(CISC)和精简指令集计算机(RISC)工作站;按显示器可分为单色和彩色工作站;按工作站有无磁盘配置可分为有盘工作站和无盘工作站;按照工作站的机箱形式和大小可分为桌边型和桌上型工作站。此外,还可按照工作站的性能和价格来分类。

工作站的应用领域十分广泛,例如:科学和工程计算、软件开发、计算机辅助分析、计算机辅助制造、计算机辅助工程、工程设计和应用,图形和图像处理,办公、金融和商业事务处理,过程控制和信息管理系统等。

参考文献

张树增等. 工作站——一种新型的计算机. 北京: 电子工业出版社, 1992 (张学孝 李英敏)

gonggaoban xitong

公告板系统(bulletin board system, BBS)

在计算机网络中,为用户提供一个寄存邮件、读取通告、参与讨论和交流信息的环境的信息系统。

公告板系统的构成和运转 公告板系统(BBS)的硬件常是微型计算机,也可以是工作站或小型计算机,并带有较大容量的硬磁盘。它一般通过调制解调器与电话线相连接。BBS 的软件通常是特殊编制的,除通信功能外,还可进行用户管理等。

BBS 的日常运转事务由系统操作员负责。BBS 可以是公用的或专用的。其会员是公众或经过资格审查的会员。

公告板系统的功能和服务 它有以下几方面:

(1) 通告 BBS 的公众消息包括各种联机布告。BBS 一般只有小型的电子邮件系统。很少有实时会议,但许多 BBS 允许系统操作员与某用户实时交谈。

(2) 特殊兴趣组(SIG) 某专题的相关内容的汇集,也提供主机的主要服务内容,即公告栏和文件传送。其内容十分丰富,通常由 SIG 选单来引导进入。

(3) 文件传送 大多数 BBS 除了支持消息交换,还支持软件的复制。因此,它至少要支持 ASCII 和 XModem(二进制文件传送用)这两种传送方式。存储文件用的数据库通常包含简单的目录搜索。

(4) 其他服务 一些 BBS 提供联机游戏、图形等功能,对电子邮件和文件管理也有所增强。

公告板系统的结构 BBS 的布告栏和文件传送区域通常用主题和子标题进行组织。其主题常反映 BBS 运转者的兴趣和口味。用户按选单引导进入各个主题。

公告板系统的意义及前景 布告栏系统提供给用户交流信息的手段,特别是面向主题的特点使其能有效地获取所需要的信息,因而有较广泛的发展前景。

(史美林)

gonggong guanli xinxi xieyi

公共管理信息协议(Common Management Information Protocol, CMIP)

由国际标准化组织 ISO 为开放系统互连(OSI)制定的网络管理协议。该协议在 ISO 的国际标准 ISO 9596 中给出了规范。国际电信联盟电信标准部也有相应的 ITU-T X.711 建议。在 ISO 的 OSI 体系结构中,协议规范总是和服务定义相配合的。与公共管理信息协议

(CMIP)相配合的是公共管理信息服务(CMIS),这些服务是在ISO 9595中定义的,相应的是国际电信联盟电信标准部的建议ITU-T X.710。

在网络管理系统中,管理者和被管资源的代理都是公共管理信息服务的用户(参见网络管理)。CMIS提供了三类服务:管理联系、管理通知和管理操作。CMIS是由CMIP机支持的。双方的CMIP机通过执行CMIP交换协议数据单元(PDU)来实现公共管理信息服务。CMIP中共有11种PDU,它们是:m-Event Report、m-Event Report-Confirmed、m-Get、m-Line-Reply、m-Set、m-Set-Confirmed、m-Action、m-Action-Confirmed、m-Create、m-Delete和m-Cancel-yet-Confirmed。每个协议数据单元(PDU)都带有来自CMIS的定义参数、结果参数和错误参数。从这些PDU的名称可以大致猜测其含义,如Event Report表示报告非正常的事件,Set表示修改对方的变量、Get表示收集对方的信息、Action表示发命令到对方以执行某种操作、Create表示创建对象、Delete表示删除对象等。其中,m代表这些都是管理用的协议数据单元,而带有Confirmed的PDU都是对不带Confirmed的同样的PDU的证实性回答。

另一个更为著名的网络管理协议是Internet中采用的简单网络管理协议(SNMP)。CMIP是在SNMP基础上设计的,并试图克服SNMP早期版本的一些缺陷(如安全性方面),具备更为灵活和强大的功能。如前所述CMIP有11种PDU,而SNMP中只有5种PDU。SNMP中虽然也使用了对象的概念,但实际上只有数据属性;CMIP中的对象则不仅有数值,而且有行为,真正体现了面向对象技术的许多特点。CMIP得到许多国家的政府和大公司的支持,曾经有人认为它将在近期取代SNMP,事实并非如此。一个重要的原因是,CMIP过于复杂,不但实现时要投入大量人力和物力,在运行时也要耗费大量的计算机和网络的资源。

参考文献

高传善等.数据通信与计算机网络.北京:高等教育出版社,2000
(高传善)

gongli yuyi

公理语义 (axiomatic semantics) 运用数学中的公理化方法给出的计算机语言的语义。不同的人了解程序的含义时有不同的要求。例如,有的人只关心程序的数据输入和输出,而不关心程序是否正确终止。公理语义就是研究如何将这些不同的

要求形式化,并根据这些要求严格给出程序设计语言的语义。

1967年美国R. W. Floyd提出描述人们所关心的程序含义,以及如何去论证一个程序是否具有某种含义的数学方法,1969年英国C. A. R. Hoare首次用公理系统定义了一类程序设计语言的语义。1975年荷兰E. W. Dijkstra提出基于最弱前置条件的公理语义描述方法。

在定义语言的公理语义时,必须先给出描述所关心的程序语义的形式化方法,然后建立公理系统,规定语言成分的有关语义。如果用一个程序 P 去计算自然数的阶乘,这个程序中的变量 x 在程序开始执行时,存放用户输入的自然数值 k ;而在程序执行终止时,存放要输出的结果。用户关心的是程序 P 计算的结果值是否确是输入值的阶乘。在公理语义中,使用公式 $\{x=k\}P\{x=k!\}$ 表示程序 P 的这一部分含义;若 P 执行前 x 的值等于 k ,则 P 执行完毕后 x 的值等于 $k!$ 。程序 P 执行前的条件 $\{x=k\}$ 称为 P 的前置条件,执行后的条件 $\{x=k!\}$ 称为 P 的后置条件。这类公式称为归纳命题。一般地说,归纳命题用 $\{R\}P\{Q\}$ 表示;若程序 P 执行前,其程序变量的值满足前置条件 R ,则程序 P 执行完毕后,其程序变量的值满足后置条件 Q 。归纳命题用来作为描述程序语义的工具,公理语义就是用归纳命题的公理系统来定义程序语言的语义。

执行赋值语句($x:=e$)的结果是将程序变量 x 的值变为执行该语句前表达式 e 的值。也就是说,执行该语句后 x 的值等于执行该语句前表达式 e 的值。因此,若表达式 e 在语句($x:=e$)执行前满足条件 R ,那么程序变量 x 在语句($x:=e$)执行完毕后亦应满足条件 R 。故归纳命题 $\{R[e/x]\}x:=e\{R\}$ 应该永远成立。其中 $R[e/x]$ 成立,表示将 R 中的 x 代为 e 后, R 成立,即 e 满足 R 。在公理系统中,公理是一种永远成立的命题。这样采用 $\{R[e/x]\}x:=e\{R\}$ 作为公理,就表达了语句($x:=e$)的语义,使用不同的 R ,可反映不同的用户对赋值语句语义的要求, R 可以只涉及输入输出变量;也可以涉及有可能产生副作用的其他变量。

执行顺序语句($s_1;s_2$),就是使用执行($s_1;s_2$)前程序变量的值,先执行 s_1 ;然后使用执行 s_1 后程序变量的结果值,再执行语句 s_2 , s_2 执行完毕后的程序变量值,就是执行顺序语句($s_1;s_2$)的结果值。故若执行($s_1;s_2$)前的程序变量值满足条件 R ,($s_1;s_2$)执

行完毕后的值满足 T , 那么就可找到关于中间结果的条件 Q , 使得若 R 为 s_1 的前置条件, 则 Q 为 s_1 的后置条件, 而且以 Q 为 s_2 的前置条件, 则 T 就是 s_2 的后置条件, 这样就可以采用推理规则

$$\frac{\{R\}s_1\{Q\}, \{Q\}s_2\{T\}}{\{R\}(s_1;s_2)\{T\}}$$

来规定顺序语句的语义。公理系统中的推理规则表示当横线上方的命题都成立时, 则横线下方的命题亦成立。人们可使用不同的 R, T 来表示自己所了解的有关语义。

其他语言成分的公理语义也是用公理和推理规则类似地给出, 但有的成分(如过程调用等)的语义, 比起上述语句的语义要复杂得多。

论证一个程序是否具有某种含义的过程和论证一个程序是否具有某种特性的过程是完全一致的。故公理语义学是程序正确性研究的理论基础。程序验证的研究也进一步促进公理语义学的发展。

寻求适用于描述程序语义, 且便于语义推导的逻辑语言是公理语义学研究的一个重要方面。20 世纪 70 年代出现了使用时态逻辑来定义语言的语义, 称为时态语义。另外如动态逻辑、算法逻辑在语义学中的应用, 也都在发展之中。

参考文献

周巢尘. 形式语义学引论. 长沙: 湖南科技出版社, 1985

(李晓山 周巢尘)

gongyong jiaohuan dianhuawang

公用交换电话网 (public switched telephone network, PSTN)

向公众提供电话通信服务的一种通信网。它是国家公用通信基础设施之一, 一般由国家邮电部门统一建设、管理和运营。

根据地理范围, 公用交换电话网(PSTN)可分为: 国际长途电话网、国内长途电话网、本地电话网以及用户延伸或补充设备。

公用交换电话网主要提供电话通信服务, 同时还可提供非话音(言语)的数据通信服务, 例如电报、传真、数据交换、形象图文等。

交换机是公用交换电话网的核心设备。由存储程序控制的交换机称为**程控交换机**, 它将各种控制功能、步骤、方法等编成程序, 放入存储器, 以此来控制交换机工作。如果传送和交换的是模拟话音(言语)信号, 则称为**程控模拟交换机**; 如果传送和交换的是数字话音(言语)信号, 则称为**程控数字交换机**。在 20 世纪 60 年代, 各国竞相研制程控数字交

换机, 1970 年法国成功地开通了世界上第一个程控数字交换机系统 E10。从此, 电话交换技术从传统的模拟交换时代进入了数字交换时代, 而且对电话网开通了非话音(言语)业务, 进而为实现综合业务数字交换奠定了基础。

参考文献

汪润生, 周师熊编著. 数据通信工程. 北京: 人民邮电出版社, 1990

(史美林)

gongyong shuju wang

公用数据网 (public data network, PDN)

一种向公众提供数据通信服务的通信网。它是国家公用通信基础设施之一, 一般由国家统一建设、管理和运营。

公用数据网(PDN)由交换结点(即**交换机**)、网控中心、用户入网设备、通信线路等设施组成。按照全网统一的编址方案, 每个入网用户可与网上其他用户通信。公用数据网只负责数据从发送端到接收端的透明的无差错传输, 用户之间通信的高层协议或应用业务则由用户自己协商和选择。公用数据网实际上是一个提供公用数据通信服务的通信子网, 而各用户或用户组织借助通信子网提供的服务可以组建自己的**虚拟专网**和**信息系统**。

在公用数据网出现以前, 世界各国都已建立了**公用交换电话网**。在组建公用数据网时, 往往采用与公用交换电话网相结合的方针, 利用原有的电路交换线路, 通过**调制解调器**将交换机、网控中心和用户入网设备互连起来进行数据通信。

20 世纪 70 年代中期, 世界上已出现了一些公用数据网, 例如美国的 TELENET(1975 年)、TYMNET(1977 年), 加拿大的 DATAPAC(1977 年), 法国的 TRANSPAC(1978 年)等。后来, 许多国家都建立了公用数据网作为数据通信用的国家信息基础设施。我国于 1989 年 11 月建成国家公用数据网, 取名为 CHINAPAC(CNPAC)。

根据数据信息在公用数据网内交换方式和相应的交换技术发展状况, 公用数据网有**公用数字数据网(DDN)**、**公用分组交换数据网(PSDN)**、**公用帧中继网**、**公用 ATM(异步传送模式)网**等之分。公用数据网除了向公众提供一般的数据交换服务平台外, 还提供公用电子信箱业务、公众宽带多媒体业务、公用电子数据交换(EDI)业务、公众互联网业务等, 无线公用线数据通信网及相应的业务也在发展中。

(史美林)

gongyao jichu sheshi

公钥基础设施 (public key infrastructure, PKI) 在分布式计算系统中提供的使用公钥密码系统(参见密码学)和 X. 509 证书安全服务的基础设施。公钥基础设施 (PKI) 产品和服务允许使用者在网络上建立一个安全领域,在该领域中可以签发密钥和证书。PKI 支持使用者在建立的安全领域中进行加解密和证书的使用、管理以及安全政策管理等。密钥管理包括密钥的更新、恢复和托管,证书管理包括证书的产生和撤销。PKI 还提供通过证书

层次结构或者通过直接交叉证书的方法在本地安全领域与其他安全领域之间建立相互信任的关系。

图 1 表示了 PKI 的体系结构。除了证书以外,PKI 还包括其他几个组成成分。PKI 最基本的组成是证书的主体。它通常是用户,也可以是任何拥有公钥的一个公司、组织、系统或者应用。例如万维网 (Web) 站点就可以成为证书主体。它通过安全套接层 (SSL) 或者其他协议与浏览器建立安全通信信道。用户和应用软件系统可以成为证书的客体,这样,它们和其他实体将是证书的使用者。

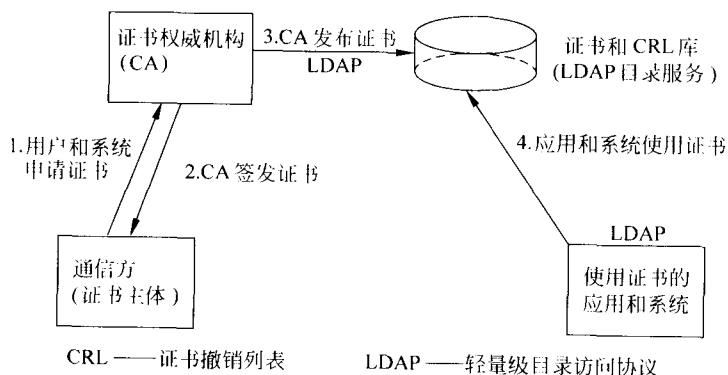


图 1 PKI 体系结构

证书权威机构 (CA) 创建并签发证书。通常一个 CA 将为一个有限的用户团体签发证书,这样的用户团体通常被称为安全领域。CA 还维护并且发布证书撤销列表 (CRL)。当证书以及证书中的公钥失效时通常采用证书撤销列表 (CRL) 这种集中方式通知用户和应用。CA 通常将 CRL 发布在目录服务的某个位置甚至某个特定的统一资源定位地址 (URL) 上。

目录系统是 PKI 的重要依靠,目前支持简便的目录访问协议 (LDAP) 是最基本的要求。因为 PKI 将使用目录访问协议来存放、发布、查找和获取密钥,如果目录访问协议不支持基于证书的对象类和属性的话,PKI 产品通常将涉及对目录访问协议的扩展。由于 LDAP 和 X. 509 是目前占优势的目录访问协议标准,大多数 PKI 产品默认支持 X. 509 (X. 520, X. 521 和 X. 509) 定义的证书对象类和属性。

由于建立公钥基础设施 (PKI) 的目标是需要与其他网络以及 Internet 互连,以便在全球范围内实现电子商务和安全通信等应用。因此,PKI 必须基于国际标准以保证它的互操作性,这是实现 PKI 的

最低限度的要求。但是目前 PKI 体系结构和标准才推出,现有的很多证书服务器产品,例如 Netscape 的 Certificate Server 和 Microsoft 的 Certificate Server,只涉及了证书的生成和撤销,并没有解决密钥等其他问题,因此没有完全解决企业对于公钥体系结构的需求,并不是一个完整的 PKI 的解决方案。

参考文献

CCITT Recommendation X. 509. The Directory - Authentication Framework, 1988 (胡道元)

gongyao mima jishu

公钥密码技术 (public key cryptography)

在密码体制中加密和解密采用不同的两个相关的密钥的技术。又称不对称密钥技术。公钥系统的概念是 Diffie 和 Hellman 在 1976 年提出的。目前公钥算法有很多种,共同特点是:每个通信方在进行保密通信时有两个相关的密钥,其中一个公开,另一个保密。公钥算法比传统密钥算法(参见密码学)计算复杂度高,大量数据加密时传统加密算法的速度比公钥加密算法快 100 ~ 1 000 倍,因此公钥算法常被

用来对少量关键数据(例如传统加密算法的密钥)进行加密,或者进行**数字签名**。常用的公钥算法有:①由 Rivest、Shamir 和 Adleman 提出并以他们名字首字母命名的 RSA 算法,它可以实现加密和数字签名功能;②El Gamal 和数字签名标准(DSS)的算法实现了签名功能,但是没有加密功能;③Diffie - Hellman 算法用于建立共享密钥,既没有签名功能,也没有加密功能,一般与传统密码算法共同使用。这些算法复杂度各不相同,提供的功能也不完全一样。

使用最广的公钥加密算法是 RSA。RSA 使用两个密钥,一个公共密钥,一个专用密钥。如用其中一个加密,则可用另一个解密。密钥长度从 40 位到 2 048 位可变。RSA 算法在加密时把明文分成块,块的大小可变,但不能超过密钥的长度。然后再把每一块明文转化为与密钥长度相同的密文块。密钥越长,加密效果越好,但加密、解密的开销也大,所以要在安全与性能之间折衷考虑,一般 64 位是较合适的。RSA 的一个比较知名的应用是安全套接字层(SSL)。在美国和加拿大 SSL 应用 128 位密钥的 RSA 算法。

公共密钥的优点在于,也许你并不认识某一实体,但只要你的服务器认为该实体证书权威(CA)是可靠的,就可以进行安全通信,而这正是 Web 商务这样的业务所要求的,例如信用卡购物。服务方可根据客户的 CA 的发行机构的可靠程度来决定自己的资源授权。

公钥方案较对称密钥方案(参见**私钥密码技术**)处理速度慢,因此,通常把公钥与对称密钥技术结合起来,以实现最佳性能。即用公钥技术在通信双方之间传送对称密钥,而用对称密钥来对实际传输的数据加密、解密。另外,公钥加密也用来对对称密钥进行加密。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999
2. 卢开澄. 计算机密码学. 北京:清华大学出版社,1998 (胡道元)

gongneng guiyue

功能规约 (functional specification) 软件所要完成功能的准确而完整的陈述。它描述的是软件要做什么以及只做什么。书写功能规约的语言称为**功能规约语言**,它可以是自然语言,相应的规约称为非形式化功能规约;也可以是专门设计的形式功能

规约语言或**广谱语言**,相应的规约称为**形式功能规约**。功能规约一方面作为软件开发者与用户之间的契约,另一方面又是软件设计与实现者的依据。

一个好的功能规约必须逻辑上不矛盾,即满足**一致性**;它应当正确、完整地反映用户对软件的功能需求,即满足正确性或完备性;它不应当产生多种不同的理解,即没有歧义性;它还应当简明,易于理解,易于实现等。

在软件设计前先写出它的功能规约体现了功能抽象和分解的基本原则。通常,一个功能规约将软件所要做的事从软件开发的角分成若干个相对独立的功能块,而功能块的划分应既考虑到逻辑清晰性,又考虑实现的方便性。每一块都通过描述对于给定的输入应得到的输出给出这一块“做什么”的陈述,而具体“如何做”在规约中一般不给出,软件设计者可根据功能块的要求自行选择实现方法。在依据规约设计、实现程序之前,要对规约的有效性进行检查,即检查规约的一致性、完备性和无歧义性。而检查一个程序是否满足它的规约的工作称为程序正确性验证。

非形式化功能规约易读,但有效性和程序正确性的检查困难,在简明性和易实现性等方面也不易保证。对具有一定数学基础的人来说,形式功能规约有许多优越性(参见**形式功能规约**)。(伊波)

gongneng guiyue yuyan

功能规约语言 (functional specification language) 参见**功能语言**。

gongneng yuyan

功能语言 (functional language) 用以书写软件**功能规约**的语言。软件功能规约是软件所要完成功能的精确而完整的陈述;它描述的是软件要做什么,以及只做什么。功能规约是需求定义的功能抽象,它对需求定义中用户所需的功能进行重新组织和分块,使其表述为(接近于)数学模型的形式,以此作为软件设计与实现的依据。功能语言通常又称为**功能规约语言**。

在软件工程发展早期,软件功能主要采用非形式的自然语言加以描述。这样的非形式功能规约具有易书写、易理解和易使用的特点,一般用户均会使用;但由于自然语言的歧义性、模糊性和不完备性而导致难以开展软件的形式化和自动化方法研究。为

了提高软件生产率与软件产品可靠性,出现了形式软件功能规约语言。这种语言具有良好的数学基础,易于研究软件规约的各种性质和相互间的语义关系,如一致性、完备性、等价性和精化关系等,不仅避免了自然语言的歧义性、模糊性和不完备性,而且奠定了能保证程序正确性的各种形式化方法如 WP 和 VDM 等方法的基础。不仅如此,形式功能规约语言还使得软件自动化的实现成为可能。因此,形式功能规约语言逐渐为人们所接受并应用于软件工程实践。

从形式化的角度看,功能语言可分为非形式规约语言和形式规约语言。非形式规约语言是指未加限定的自然语言,而形式规约语言是指其语法和语义均显式精确定义的语言。由于软件功能的形式化描述是计算机科学中十分重要且相对成熟的领域,下文将集中讨论形式功能规约语言。从理论基础的角度看,功能规约语言可分为代数类语言和逻辑类语言。代数类语言系指以异调代数、范畴论等代数理论为主要理论基础的规约语言,如 OBJ, CLEAR 等;逻辑类语言系指以一阶谓词演算等逻辑理论为主要理论基础的规约语言,如 Z, VDM-SL 等。当然,有些语言将这两个途径有机地结合起来,例如, Larch 语言族中每一语言包括共享语言和接口语言两部分。其中共享语言基于代数方法,而接口语言则基于逻辑方法。此外,还有一类广谱语言。广谱语言中不仅包含功能规约机制,还含有设计规约等较低级的成分。

功能语言主要涉及规约对象、规约方法、以及规约性质等。

规约对象主要包括抽象数据(数据抽象)和抽象过程(过程抽象)两类。它们分别反映了对软件系统中所涉及的数据以及相应的处理进行抽象的结果。抽象数据区别于数据的本质特征在于它只涉及数据及其运算的性质而与具体实现无关;而抽象过程区别于过程的本质特征在于它只刻画相应处理“做什么”的外部行为而与内部的实现细节无关。所谓抽象过程是指从输入值集到输出值集的映射;抽象过程规约的主要是如何刻画抽象过程“做什么”的功能而不涉及“如何做”的具体算法与实现功效。抽象数据通常由抽象数据类型来反映。抽象数据类型是封装原理和信息隐蔽原理的集中体现;其本质特征在于与数据具体表示无关。因此,抽象数据类型规约的关键问题是如何体现这一特征。

规约方法主要研究如何对抽象过程与抽象数据

类型进行规约。**前后断言方法**是一种常用的对抽象过程进行规约的方法。它通过给出一对基于一阶谓词演算的断言,来刻画抽象过程输入与输出的性质及其相互间的关系,以此来描述抽象过程的功能而与具体算法无关。采用前后断言方法,抽象过程的功能规约由两部分组成,其一是接口描述,它刻画了抽象过程与外界的接口,主要包括抽象过程的名、输入、输出以及输入输出的取值范围。其二是功能描述,它刻画了抽象过程“做什么”的功能,主要包括前断言和后断言两部分。一般说来,前后断言方法具有良好的数学基础,易于研究规约的性质和相互间的关系,如规约的一致性、完备性和精化关系等。但是,前后断言方法也有局限性,它受限于所能使用的谓词。

抽象数据类型规约的主要方法有二:其一是代数方法,即用等式公理来直接刻画抽象数据类型中运算的性质而与具体表示无关;其二是模型方法,即通过给出抽象数据类型的抽象模型来达到规约的目的。采用代数方法,抽象数据类型的规约由两部分组成,一是语法部分,二是公理部分;语法部分给出了抽象数据类型的名以及它所提供的操作的定义域和值域,同时,它也给出了与之相关的其他类型名。公理部分则通过给出一组刻画各操作性质的方程作为公理来定义各操作的含义。从语义的角度看,代数规约的语义模型有多个且形成一个谱,谱的两端分别称为始语义模型和终语义模型;从而为相应的实现提供了灵活性。归结起来,抽象数据类型的代数方法能够完整地刻画其运算的性质而与表示无关,从而较好的体现了抽象数据类型的特点。然而,它也有一定的局限性,对于某些简单的数据类型,要给出其代数规约十分困难。一个著名的例子是 M. J. Majster 提出的遍历栈。对于这样一个简单的数据类型,如果要使用代数方法,则要么使用隐含的辅助函数,要么使用无穷多的公理,结果均难令人满意。

与代数方法不同,模型方法不是对抽象数据类型中运算的性质加以直接刻画,而是通过某些已知的具有良好数学性质的(抽象)数据类型来给出所要定义的新类型的抽象模型,用已知类型运算的性质来间接刻画要定义的数据类型中运算的特性,从而达到定义新的抽象数据类型的目的。一般说来,抽象数据类型的模型规约由以下三部分组成:

- (1) 状态集的定义(可能包含不变式);
- (2) 初始状态定义(通常只有一个);

(3) 运算集的定义,通常采用输入输出断言刻画。

模型只能理解成行为的描述而与具体实现无关;但如果不加注意,模型规约可能引入与实现有关的内容。这种现象称为实现斜偏。

由于形式规约语言具有较好的数学基础,因此,易于研究由它所书写的形式规约的性质和相互关系,从而为形式规约的正确性验证和软件的形式化与自动化开发奠定了良好的基础。一般说来,规约性质包括一致性、完备性、等价性、精化关系等;其中一致性和完备性是为了保证软件规约在某种意义上是正确的;而等价性和精化关系通常用以保证软件开发过程的正确性。同一类性质在不同类型的规约结构中常常有不同的含义与表现形式。例如,就完备性而言,它在抽象过程功能规约中的含义不同于它在抽象数据类型代数规约中的含义。

概括起来,功能语言(特别是形式化功能规约语言)的研究取得了较大进展,其理论基础日臻完善,方法日趋成熟,并逐步应用于软件工程实践。进一步的工作包括它在大型软件开发中的应用以及基于形式化功能规约语言的形式化与自动化方法的研究。

参考文献

1. 徐家福,陈道蓄,吕建,王志坚. 软件自动化. 北京:清华大学出版社,南宁:广西科学技术出版社,1994

2. Gehani N and McGettrick A (ed). Software Specification Techniques. Addison-wesley Publishing Company, 1986 (吕建)

gongying guocheng

供应过程 (supply process) 供应者为获取者提供软件产品的一系列活动。它从理解系统或软件产品的需求开始,经过准备投标、签定合同、制订计划、实施和控制、评审和评价等活动,直至交付完成。供应者是那些提供软件产品的机构。

供应者要对供应过程进行管理(参见管理过程);并结合项目对供应过程的各项活动进行剪裁(参见剪裁过程)。

供应过程主要包括以下各项活动:

(1) 开始 供应者要对项目招标书中的系统需求等进行认真研究,并结合自身的方针和其他因素对照分析,决定是否投标或接受合同。

(2) 准备投标 为响应招标作好准备并编写一

份投标书。

(3) 签订合同 供应者就向获取者提供软件产品与他们进行有关合同的谈判,并签订合同;供应者可以请求修改合同,并以此作为改变控制机制的一部分。

(4) 制定计划 该活动有下列各项具体任务:

①供应者要对合同中所规定的系统需求进行全面分析,从而确定一个机构,以用于管理和确保完成该项目,以及对要交付的软件产品进行质量保证。②如果在合同中未具体指明,那么供应者要选用与项目的范围、规模和复杂程度相匹配的软件生存周期的各有关过程和活动。③确定计划要求,其中最好要包括对资源的要求和获取者的适当参与等。④根据风险大小等因素,对开发软件产品方案进行选择,一般来说有以下几种选择方案:使用内部资源开发;以子合同方式开发;从内部或外部来源取得现货产品;上述三种方案的结合。⑤制定项目管理计划,并将其写成文档,文档中一般要包含如下内容,但不仅局限于这些:项目进行过程中每个组织机构(包括外部机构)的责任和权利;应用于开发、运行和维护的工程环境,其中包括测试环境、程序库、设备、仪器设施、标准、规程和工具等;选用的有关过程和活动的工作细目,其中包括软件产品、不交付的软件、文档、预算、人力物力资源、软件规模及任务进度等;软件质量管理和保证的要求,必要时要另行制定质量保证计划;系统安全、保密和其他关键需求的管理,必要时要另行制定安全和保密计划;如果要建立分包合同,则要对分包合同的供应者进行管理,其中包括对他们的选择以及他们和主合同获取者的参与等;质量保证(参见支持过程);验证和确认(参见支持过程),一般还要包括与验证和确认机构的联系、结合等方法;按合同要求,由获取者参与的联合评审、审计(参见支持过程),以及和获取者的非正式会面、报告、修改及其实施、验收等;风险管理,即涉及潜在技术、成本和进度诸风险因素的管理;保密方针,即在有关结构层次上存取信息的准则和规定;管理上要求的批准、专有权利等;对计划、跟踪和报告的方法;人员培训。

(5) 实施和控制 该活动中供应者要落实和执行以上制定的项目计划,并依照开发过程中有关活动来开发软件,依照运作过程中有关活动对其运作,依照维护过程中有关活动内容对该软件产品进行维护,在整个项目计划实施过程中,供应者应监督和控制软件产品的开发进展和质量,这是一个连续且反

复执行的任务,具体内容为监督技术性能、成本、计划、项目开展情况报告;发现问题、记录和分析解决问题;若有分包合同,则要对分包合同的供应者进行管理(参见管理过程),并向他们传达必要的主合同要求,以保证交给主合同获取者的软件产品均能符合必要的主合同要求;供应者还要按照合同和项目计划的规定与独立的验证、确认、测试机构及其他有关各方进行交接和联系。

(6) 评审和评价 在适当时机,供应者要按合同开展评审活动,并与获取者加强联系,如支持非正式会面、验收评审、验收测试、联系评审和审计等。另外,供应者还要进行软件产品的验证和确认,以表明软件产品满足系统需求等各项目要求,并向获取者提供关于评价、评审、测试和问题解决的报告,需要时还要进行质量保证等其他有关活动。

(7) 交付和完成 按合同规定向获取者交付软件产品并就该产品向他们提供必要的支持。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes-IEEE Std 1074 ~ 1991

2. ISO /IEC 12207:1995 Information Technology-Software-Part 1: Software Life-Cycle Process

(陈森芬)

gongxiang cunchu

共享存储(shared memory) 两个或多个处理机共用一个主存储器的并行体系结构。每一个处理机都可以把信息存入主存储器,或从中取出信息。处理机之间的通信通过访问共享存储器来实现。

共享存储计算机系统由于支持传统的单地址编程空间,减轻了程序员的编程负担,因此它具有较强的通用性,且可以方便地移植现有的应用软件。早期的共享存储计算机采用集中式的共享存储器,即多个处理机通过总线、交叉开关或多级互连网络等与共享存储器相连,所有处理机访问存储器时都有相同的延迟。然而,由于多个处理机共享存储器,使得存储器成为系统瓶颈。为此,在规模较大的共享存储计算机系统中,把共享存储器分成许多模块,并使它们分布于各结点之间(一个结点可能有一个或多个处理机),这种系统称为**分布式共享存储(DSM)**计算机系统,即每个结点包含共享存储器的一部分,结点之间通过可伸缩性好的互连网络相连。分布式的存储器和可伸缩的互连网络增加了访存带宽,但导致不均匀的访存延迟。

为了缓解由存储共享引起的冲突以及由存储器分布引起的访存延迟,共享存储计算机系统的处理器中一般都有高速缓冲存储器。但高速缓冲存储器的使用带来了高速缓冲存储器一致性问题(参见**高速缓冲存储器一致性**)。另外,存储器的分布使处理机访问不同的存储单元有不同的延迟。为了保证并行程序的正确执行,需要有某种高速缓冲存储器一致性协议和存储一致性模型。

高速缓冲存储器一致性协议是把系统中的一个处理机新写的值传播给其他处理机的机制。高速缓冲存储器一致性协议都是为实现某种存储一致性模型而设计的。

存储一致性模型是系统设计者和程序员之间的一种约定,它给出了判断共享存储程序及结构正确的标准,其中顺序一致性模型被普遍作为共享存储程序执行正确的标准,也是定义其他弱一致性模型的基础。在顺序一致性模型中,多个处理器并行执行程序的结果等于把每个处理机所执行的指令流按某种方式顺序地交织在一起在单机上执行的结果。如果在多处理机环境下的一个并行执行的结果和同一程序在单处理机多进程环境下的执行结果相同,则此并行执行正确。

在共享存储系统中,为了实现顺序一致性模型,需要对访存事件次序施加严格的限制,为了放松对访存事件次序的限制,人们提出了一系列弱存储一致性模型。这些弱存储一致性模型的基本思想是:在顺序一致性模型中,虽然为了保证正确执行而对访存事件次序施加了严格的限制,但在大多数不会引起访存冲突的情况下,这些限制是多余的,因此可以让程序员承担部分执行正确性的责任,即在程序中指出需要维护一致性的访存操作,系统只保证在用户指出的需要保持一致性的地方维护数据一致性,而对用户未加说明的部分,则可以不考虑处理机之间的数据相关。

根据存储器的分布、一致性的维护以及实现方式等特征,常见的共享存储系统的体系结构有以下几种:

(1) 无高速缓冲存储器的集中式共享存储结构

这种结构的处理机没有高速缓冲存储器,多个处理机通过交叉开关或多级互连网络等直接访问共享存储器。由于任一存储单元在系统中只有一个备份,这类系统不存在**高速缓冲存储器一致性问题**,系统的可伸缩性受限于交叉开关或多级互连网络的带宽。采用这种结构的典型例子是并行向量机及一些

大型机,如美国 Cray 公司的 Cray-XMP、YMP-C90 等。

(2) 基于高速缓冲存储器的集中式共享存储结构 在这种结构的系统中,每个处理机都有高速缓冲存储器,多个处理机一般通过总线与存储器相连。每个处理机的高速缓冲存储器通过侦听总线来维持数据一致性。由于总线是独占性资源,这类系统的伸缩性是有限的。这种结构常见于采用对称式多处理机(SMP)系统(参见并行处理系统)的服务器和工作站中,如 4DEC, SUN, Sequent 以及 SGI 公司的多机工作站产品。

(3) 具有高速缓冲存储器一致性的分布式共享存储结构 这种结构称为高速缓冲存储器一致的非均匀存储访问(CC-NUMA)结构。这类系统的共享存储器分布于各结点之间。结点之间通过可伸缩性好的互连网络相连,每个处理机都能缓存共享单元,高速缓冲存储器一致性的维护是这类系统的关键,决定着系统的可伸缩性。常采用基于目录的方法来维持处理机之间的高速缓冲存储器一致性。这类系统的例子有 Standford 大学的 DASH 和 FLASH, MIT 的 Alewife, 以及 SGI 的 Origin 2000 等。

(4) 唯高速缓冲存储器的分布式共享存储结构(COMA) 在这种结构中,每个结点的存储器相当于一个大容量的高速缓冲存储器,数据一致性也在这一级维护。这种系统的共享存储器的地址是活动的。存储单元与物理地址分离,数据可以根据访问模式动态地在各结点的存储器间移动和复制。其优点是当处理机的访问不在高速缓冲存储器命中时,在本地共享存储器命中率较高。其缺点是当处理机的访问不在本结点命中时,由于存储器的地址是活动的,需要一种机制来查找被访问单元的当前位置,因此延迟很大。采用这种结构的系统有美国 Kendall Square Research 公司的 KSRI 和瑞典计算机研究院的 DDM。此外,这种结构常用于共享虚拟存储系统中。

(5) 无高速缓冲存储器一致性的分布式共享存储结构 这种结构称为无高速缓冲存储器一致性的非均匀存储访问(NCC-NUMA)结构。它的特点是虽然每个处理机都有高速缓冲存储器,但硬件不负责任维护高速缓冲存储器一致性,而由编译器或程序员来维护。其典型代表是 Cray 公司的 T3D 及 T3E 系列产品,在 T3D 和 T3E 中,系统为用户提供了一些用于同步的库函数,便于用户通过设置临界区等手段来维护数据一致性。这样做的好处是系统伸缩性强,高档的 T3D 及 T3E 产品可达上千个处理机。

(6) 共享虚拟存储结构 这种结构又称为软件分布式共享存储结构。它的基本思想是在基于消息传递(参见并行处理系统)的大规模并行处理系统或集群式计算系统中,用软件的方法把分布于各结点的多个独立编址的存储器组织成一个统一编址的共享存储空间。这种结构具有共享存储系统的可编程性和消息传递系统的硬件简单的优点,其主要问题是通信开销很大。在某些高性能计算机系统中,在结点内部利用对称多处理机系统提供硬件的共享存储,在结点间由软件实现共享存储。常见的虚拟共享存储系统有 Ivy, Midway, Munin, Treadmarks 和 JIAJIA 等。

参考文献

胡伟武. 共享存储系统结构. 北京: 高等教育出版社, 2001

(胡伟武)

gouzao leixing

构造类型 (composite type) 其值由更简单的值组合而成的类型。这些更简单的值所具有的类型称为“分量类型”或“元素类型”,它们可以是简单类型,也可以是构造类型。

现有的程序设计语言提供了多种构造数据结构的方法,如元组、记录、变体、联合、数组、集合、字符串、表、树、顺序文件等等。事实上,所有这些类型是通过下述概念来构造的,它们是:

(1) 笛卡儿积 可形式地表示为

$$S = T_1 \times T_2 \times \cdots \times T_n \\ = \{ (x_1, x_2, \cdots, x_n) \mid x_i \in T_i, 1 \leq i \leq n \}$$

笛卡儿积用于构造元组和记录,如下列记录类型

record

$I_1, : T_1;$

...

$I_n : T_n$

end

其值集为 $T_1 \times \cdots \times T_n$ 。

(2) 分离并集 我们用 $S + T$ 表示集合 S 和 T 的分离并集,每一个值或选自 S 或选自 T ,此外,每一个值还附以标记以表明其选自哪个集合。可形式定义为:

$$S + T = \{ \text{left } x \mid x \in S \} \cup \{ \text{right } y \mid y \in T \}$$

即选自 S 的值标记以 left, 选自 T 的值标记以 right。

分离并集用于构造变体记录和联合类型。如变体记录

```

record
case I: T of
L1: (I1: T1);
...
Ln: (In: Tn)
end

```

其值集为 $T_1 + \dots + T_n$ 。

(3) 映射 一个映射 m 把集合 S 中的每一值映射到集合 T 中的一个值, 记作 $m: S \rightarrow T$ 。

程序设计语言中的数组实质上表示一种有限映射, 即从一个有限集合(数组的下标集)到数组的分量集合的映射, 大部分程序设计语言仅使用整型(或其子域)作为数组的下标集, 而 PASCAL 和 Ada 语言允许数组的下标集为任意离散简单类型。

例: 下列数组类型

```
type window = array [0..511, 0..255] of
```

```
0..1
```

其值为 $\text{window} = \{0, \dots, 511\} \times \{0, \dots, 255\} \rightarrow \{0, 1\}$,

若变量 W 说明为:

```
Var W: window
```

则 $W[8, 12]$ 取接到 W 的下标为 $(8, 12)$ 的分量。

映射在程序设计语言中还以函数抽象的形式出现, 函数抽象通过一个算法来实现从 S 到 T 的映射, 函数取 S 中的任意值计算出该值在 T 中的象, 所以 S 不必是有限集。

例: PASCAL 中的标准函数 odd 实现一个从 Integer 到 Boolean 的映射, 即:

```
{0 → false, ±1 → true, ±2 → false, ±3 → true, ...}
```

(4) 幂集 考虑一个值集 S , S 的所有子集的集合构成 S 的幂集, 形式地记为

$$P(S) = \{s | s \subseteq S\}$$

幂集用于构造集合类型, 在 PASCAL 语言中, 集合类型定义具有如下形式

```
set of T;
```

该集合类型的值集即为 $P(T)$ 。PASCAL 语言提供了所有基本的集合运算。

例: `type Color = (red, green, blue)`

```
Hue = Set of Color
```

类型 Hue 的值集为 $P(\text{color})$, 即 $\{\text{red}, \text{green}, \text{blue}\}$ 的所有子集的集合。

(5) 递归类型 其值由同样具有该递归类型的值组成的数据类型, 递归类型是根据其自身来定义。

一般地, 一个递归类型 T 的值集是由下列递归

等式所定义的最小解:

$$T = \dots T \dots$$

递归类型用于定义动态数据结构。

参考文献

Watt D A. Programming Language Concepts and Paradigms. Prentice Hall, 1990 (金凌紫 陈涵生)

gutai pan

固态硬盘 (solid state disk, SSD) 用大规模集成半导体存储器件作存储介质, 实现与磁盘存储器功能等效的一种数据存储设备。又称**半导体盘**。其读写过程中没有磁盘的磁头寻道与主轴旋转等机械运动, 但其数据存取仍然以块或页为单位来进行, 这一点和磁盘以扇区为单位的方式一致; 而且从管理软件或驱动程序的角度来看, 它与磁盘有类似的功能, 故沿称其为“盘”

发展简史

固态硬盘于 20 世纪 80 年代初出现在大型或巨型计算机系统中。如在 IBM 3090 系列机中作为扩展存储器, 它和主存储器之间的最大传输速率达 216 MB/s; 在 CRAY X-MP 和 Y-MP 计算机中的固态硬盘, 其容量最高可达 4096 MB, 传输速率达 1000 MB/s。

日本 Fujitsu 公司于 1985 年和 1989 年先后研制出 F 6630 与 F 6631 两种型号的固态硬盘, 它们分别采用每片 256 kb 与每片 1 Mb 的动态随机存取存储器芯片 (DRAM 芯片)。前者最大容量 256 MB, 最大传输速率 18 MB/s; 后者最大容量 1024 MB, 最大传输速率 36 MB/s, 连接光通道时为 72 MB/s。为了解决掉电时数据的易失性问题, 以上各种固态硬盘都必须采用后备电池或硬磁盘驱动器作备份。

20 世纪 90 年代初, 快可擦编程只读存储器芯片 (Flash EPROM 芯片) 开始应用, 利用它构造了不需要后备电池或硬磁盘的全半导体化的非易失性固态硬盘。1991 年开发出用 4 Mb Flash EPROM 芯片构成的格式化容量为 20.9 MB 的固态硬盘, 以后的固态硬盘绝大多数由 Flash EPROM 芯片构成。

进入 21 世纪以来, 固态硬盘得到了飞速的发展, 并广泛应用于军事、航天、航空、公安、通信和运输等领域。其主要特点如下:

(1) 容量不断增大, 传输速率不断提升。如 M-System 公司的快闪固态硬盘 (FFD), 其容量可高达 90 GB, 突发读写速率为 100 MB/s, 稳定读写速率为 40 MB/s, 访问时间小于 0.02 ms。

(2) 接口多样化以适应不同应用。有传统的

ATA 接口(俗称 IDE),适应高速传输的 Ultra-Wide SCSI、Ultra-Narrow SCSI 接口,有 LPT 并行接口,还有 PC/104 总线、PCI 总线,应用于笔记本电脑和一些专用处理器的 PCMCIA 接口,以及使用非常方便并支持热插拔的 USB 接口等。

(3) 更小型化以适应特别应用。有 2.5/3.5 英寸的标准封装产品,还有应用于 PDA、手机、数码相机、MP3 播放器和其他小型化设备的袖珍闪存(CF)卡和智能卡,以及更薄更小的 DOC 和 DOM 结构等。

(4) 更高的可靠性。如 M-System 公司的 FFD 的最小故障间隔时间(MTBF)大于 700 000 工作小时,写-擦次数大于 5×10^6 次,读次数不限;工作温度在 -40°C 至 $+85^{\circ}\text{C}$ 之间,存储温度在 -55°C 至 $+95^{\circ}\text{C}$ 之间,湿度在 5% 至 95% 之间,工作高度在海拔约 24 000 m,抗冲击、抗颤动性能良好。

(5) 更安全更智能化。如用加密协处理机直接在 DOC 内完成对称、Hash 等数据加密,以及 PKI 认证和 RSA-1024 位密钥生成等。

(6) 广泛开放于各种操作系统,如 DOS, Windows CE/XP/XPE/NT/NTE, Symbian OS, Linux, VxWorks, QNX 6.x 以及其他 OS。

基本结构

固态盘由非易失性存储器、控制器及管理或驱动软件三大部分构成。非易失性存储器是固态盘的数据存储部件,其基本结构又大致可分为两类。一类是较早使用的,由易失性随机存取存储器(RAM)和电可擦编程只读存储器或硬磁盘构成,数据的实际存取在 RAM 中完成,仅当检测到掉电或断电征兆时才执行由 RAM 至 EPROM 或硬磁盘的备份,而在重新启动时则执行由 EPROM 或硬磁盘至 RAM 的复制。另一类是目前广泛使用的,仅由 Flash EPROM 构成。采用不同的非易失性存储器,其存储子系统及控制器的结构也随之不同。

控制器是固态盘的关键部件,主要由微处理器、存储程序的只读存储器(ROM)、随机存取存储器(RAM)、数据传输控制电路、检纠错电路以及分别与主机和数据存储器两端连接的接口电路组成。另外,视不同非易失性存储器的需要,可增设电压变换、电源监控及备份控制等电路。

控制器的主要功能如下:

(1) 接收并解释来自主机的读写命令,并控制与主机的数据交换。

(2) 按一定的存储格式控制存储器的读写或擦除操作。

(3) 完成数据的检纠错计算。

(4) 管理数据文件所在的页或块的位置、擦除状态及重写次数。

(5) 对有高压要求的 Flash EPROM 进行电压变换。

(6) 对易失性存储器实施电源掉电或断电征兆的检测,并实时控制后援电源和电路的切换,完成数据备份或数据恢复。

固态盘的管理软件或驱动程序一般固化在控制器的 ROM 中,由一个微处理器执行。除了具体控制固态盘的读写及擦除外,还对页面(等价于磁盘扇区)进行管理。目前固态盘的管理方式主要有两种。一种方式是基于传统磁盘操作系统(DOS)的文件分配表(FAT),将文件以页面为单位放在固态盘中,适合用面向页面设计的 Flash EPROM 芯片来构成固态盘。这一方式的缺点是文件存不满一个页面时浪费空间。另一种方式是可安装文件系统(IFS)与快速文件系统(FFS)的结合。IFS 提供 DOS 与 FFS 驱动程序之间的接口链路。FFS 用类似栈操作的方法将文件及目录依次存入固态盘,不采用 DOS 的扇区机制,从而使无效空间减到最小。FFS 的关键之点还在于用户可以控制固态盘变满的速度,然而对那些非临时性的数据文件可减少擦除次数。然而这又需要以更小的存取单位来对 Flash EPROM 进行操作,不利于设计更高容量的 Flash EPROM。

工作原理

下面通过早期的 F 6631 固态盘和 M-System 公司的 DOC 固态盘来说明固态盘的基本工作原理。

F 6631 固态盘采用动态随机存取存储器与后备磁盘构成。整个 DRAM 最大可划分成 32 个逻辑驱动器,每个逻辑驱动器从使用角度看是一个 F 6425 磁盘。按照磁盘的柱面、磁道及扇区来划分 DRAM 地址。固态盘中所有硬部件几乎都是双份的,再加上冗余磁盘奇偶检验,使系统的容错能力有很大提高。

该固态盘的一个重要技术是非易失性存取的实现。主机实际上是对固态盘的 DRAM 进行读写,仅在断电时或启动时才对微型硬磁盘进行读写。固态盘中装有密封型铅蓄电池。一旦检测到掉电的征兆,系统立即把电源切换到蓄电池。切换后一分钟内对包括 DRAM 在内的整个固态盘供电。如果在这一分钟内电源恢复,则不改变内部状态,并从电池切换回电源。如果超过一分钟电源仍未恢复,则立即把 DRAM 中的数据全部备份到微型硬磁盘上去。

备份完成后,断开蓄电池进入真正的关机状态。备份大约需要8分钟,在此期间如果电源恢复,则仍然执行备份操作直到完毕。但最后并不进入真正的关机状态,且DRAM中的用户数据仍有效。当开机或重新合上设备的电源时,固态盘的各部分自动进入初始化,并由存储子系统把微型硬磁盘上的用户数据复制到DRAM。

图1是M-System公司2004年发布的DOC固态盘的基本结构,它由管理软件或驱动软件与DOC硬件两大部分构成。应用软件(APPS)通过操作系统

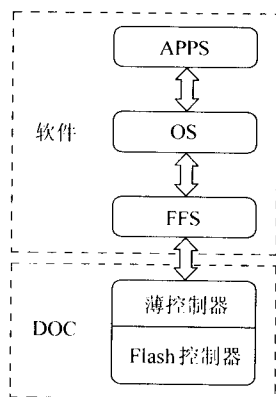


图1 DOC基本结构

(OS)和FFS调用薄控制器。图2中的薄控制器系统接口是一个等同于SRAM系统的标准8/16位接口模块,从而使开发者和系统集成者在芯片组、存储器总线开发平台或CPU方面有简单方便的选择;系统通过控制与状态模块来控制和管理工作;XIP用来解决NAND flash不便引导系统的问题;检纠错EDC/ECC硬件模块对读写中的每块(512B)数

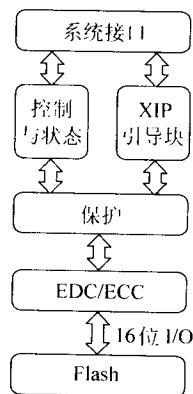


图2 薄控制器内部结构

据进行快速的二位检纠错;保护模块对Flash存储器中所有数据或代码的读写进行安全监控,用户可以定义被保护数据的大小和类型(读写),任何没有正确密钥的访问将被中止,操作也将自动失效。

性能评价

磁表面存储器(如磁盘)存在读写速度慢、易磨损、不抗震、寿命短、功耗高、体积大等弱点,而由Flash EPROM构成的固态盘在这些方面有明显优势。固态盘的数据块定位速度比磁盘快好几个数量级,前者的定位时间一般为0.1~10 μs,而后的定位时间约为4~20 ms。固态盘无机机械磨损,运转安静无噪声。它能在500 g(g为重力加速度)冲击加速度和15 g振动加速度的环境中工作(不工作时能耐1500 g冲击)。固态盘的数据可保存100年,远远超过磁盘驱动器的寿命。固态盘体积小,功耗低,一般比同容量磁盘的功耗小一个数量级,当前的DOC与DOM技术更是把固态盘的轻小型化推到了精致的层面,特别适用于各种控制和嵌入式产品。另外,DOK安全加密、热插拔、高可靠和高可用等智能化技术也使固态盘的综合性能有了很大的提高。

固态盘存在的首要问题仍然是在大容量区段内每兆字节的价格高于同容量硬磁盘。其次是一般Flash EPROM芯片的可写擦次数虽然有所提高,但目前仍维持在几十万次(部分高端产品可达几百万次以上)。另外,固态盘写入前必须进行整体或分块(几个页面)擦除,其写入速度仍待提高。采用易失性存储主体的固态盘也仍然存在电池失效和数据丢失的危险。

发展趋势

Flash EPROM固态盘的价格与其容量成正比,故努力降低其每位成本仍然是支持其发展的关键条件之一。此外,固态盘的系统结构设计中采用新的高速总线、分流和并行存取等输入输出技术,也有助于提高其I/O性能。总之,固态盘将充分发挥其高速、低功耗、高可靠性、轻薄小型化、嵌入化和智能化的特点,在低中容量区段内显示它强劲的技术和应用优势。随着Flash EPROM等技术的不断进步,它也将在大容量、低价格方面得到新的发展。

参考文献

1. 坂上好功. フラッシュメモリのソリッド・ステート・ファイルへの応用例. 電子技術, 1993; 41~45
2. <http://www.m-sys.com/Content/Products/FFDFamily.asp> (李之棠)

guzhang huifu

故障恢复 (fault recovery) 数据库管理系统中不论何时系统发生故障都能将数据库恢复到某个一致状态的机制和过程。

数据库系统中主要的故障类型包括：事务故障，即由于事务内部或外部原因造成某个事务无法继续正常执行；系统崩溃，即硬件或系统软件故障造成系统运行终止。

数据库管理系统中的故障恢复机制由两部分组成。第一，在正常事务处理时采取措施，记录数据库修改，保证有足够的信息可用于故障恢复。使用最为广泛的是日志，日志记录了数据库中的所有事务及其更新活动，包括各个事务的开始、提交、或中止，某事务对某数据元素进行了修改，其改前旧值、改后新值等。第二，故障发生后执行恢复算法，基于日志中记录的内容，将数据库恢复到某个保证数据库一致性、事务原子性及持久性的状态。为保证不论何时发生故障均能应用日志信息，系统遵循先写日志的原则，即在将任何数据库修改写到磁盘之前，必须先把相应的日志记录写到磁盘中。

此外，要支持介质故障（如磁盘故障）恢复就需要

定期地转储数据库后援副本。发生介质故障时能利用后援副本重载数据库，再利用日志来完成故障恢复。

参考文献

1. Silberschatz A, Korth Henry F, Sudarshan S 著. 数据库系统概念. 杨冬青, 唐世渭等译. 北京: 机械工业出版社, 2000

2. Date C J 著. 数据库系统导论. 孟小峰, 王珊等译. 北京: 机械工业出版社, 2000 (杨冬青)

guanjianzhen donghua

关键帧动画 (keyframe animation) 通过在关键帧处对影响画面的参数设置指定值，然后采用插值技术自动计算出其余帧参数值的运动控制技术。关键帧的概念来源于传统的卡通片制作。在早期 Walt Disney 的制作室，熟练的动画师设计卡通片中的关键画面，亦即所谓的关键帧，然后由一般的动画师设计中间帧。在三维计算机动画中，中间帧的生成由计算机来完成（图 1），插值（线性插值或样条插值）代替了设计中间帧的动画师。所有影响画面图像的参数都可成为关键帧的参数，如位置、旋转角、纹理的参数等。



图 1 关键帧与中间帧

基于关键帧的插值运动控制技术的主要步骤包括：①确定需控制的运动参数；②设置 n 个关键帧参数；③采用样条插值技术对 n 个插值点进行插值。④对该插值样条进行离散采样，求得在某一帧时的参数值。

参考文献

鲍虎军, 金小刚, 彭群生等. 计算机动画的算法基础. 杭州: 浙江大学出版社, 2000 (金小刚)

guanxi

关系 (relation) 反映集合元素之间的某种联系的集合的笛卡儿乘积的子集。

设 A 和 B 都是集合, 若 $R \subseteq A \times B$, 则称 R 为从 A 到 B 的**二元关系**。当 $\langle a, b \rangle \in R$ 时, 常用 aRb 表

示。令

$$\text{dom}(R) = \{a | a \in A \text{ 且有 } b \in B \text{ 使 } \langle a, b \rangle \in R\},$$

$$\text{ran}(R) = \{b | b \in B \text{ 且有 } a \in A \text{ 使 } \langle a, b \rangle \in R\}.$$

称 $\text{dom}(R)$ 为 R 的定义域, $\text{ran}(R)$ 为 R 的值域。如果 $R = \emptyset$, 则称 R 为空关系; 如果 $R = A \times B$, 则称 R 为全关系; 如果 $A = B$, 则称 R 为 A 上的二元关系。例如, $\{\langle n, n+1 \rangle | n \in \mathbb{N}\}$, $\{\langle n, m \rangle | n, m \in \mathbb{N} \text{ 且 } n \leq m\}$ 和 $\{\langle n, m \rangle | n, m \in \mathbb{N} \text{ 且 } n > m\}$ 均为自然数集 \mathbb{N} 上的二元关系。

设 R 为集合 A 上的一个二元关系。

自反性 若 $a \in A$, 则 $\langle a, a \rangle \in R$ 。

反自反性 若 $a \in A$, 则 $\langle a, a \rangle \notin R$ 。

对称性 若 $\langle a, b \rangle \in R$, 则 $\langle b, a \rangle \in R$ 。

反对称性 若 $\langle a, b \rangle \in R$ 且 $\langle b, a \rangle \in R$,

则 $a = b$ 。

传递性 若 $\langle a, b \rangle \in R$ 且 $\langle b, c \rangle \in R$, 则 $\langle a, c \rangle \in R$ 。

显然, 空集合上的空关系具有自反性、反自反性、对称性、反对称性和传递性, 非空集合上的空关系具有反自反性、对称性、反对称性和传递性, 但不具有自反性。自然数集合 \mathbf{N} 上的大于关系 “ $>$ ” 只具有反自反性、反对称性和传递性, 而小于等于关系 “ \leq ” 只具有自反性、反对称性和传递性。设 m 为大于 1 的正整数。若令

$$R_m = \{ \langle i, j \rangle \mid i, j \in \mathbf{N} \text{ 且 } |i - j| \text{ 是 } m \text{ 的倍数} \},$$

则 \mathbf{N} 上的二元关系 R_m 具有自反性、对称性和传递性, 但不具有反自反性和反对称性。常称 R_m 为 \mathbf{N} 上的“模 m 同余”关系, 常用 “ \equiv_m ” 表示。

设 R 为从 A 到 B 的二元关系, 则称从 B 到 A 的二元关系 $\{ \langle b, a \rangle \mid \langle a, b \rangle \in R \}$ 为 R 的逆关系, 记为 R^{-1} 。若 \bar{R} 为从 B 到 C 的二元关系, 则称从 A 到 C 的二元关系 $\{ \langle a, c \rangle \mid \text{有 } b \in B \text{ 使 } \langle a, b \rangle \in R \text{ 且 } \langle b, c \rangle \in \bar{R} \}$ 为 R 与 \bar{R} 的合成关系, 记为 $R \bar{R}$ 。

对每个集合 A , 称 A 上的二元关系 $\{ \langle a, a \rangle \mid a \in A \}$ 为 A 的恒等关系, 记为 I_A , 这时显然有

$$I_A \circ R = R = R \circ I_B, \quad R \subseteq A \times B.$$

设 A 为集合, R 为 A 上的二元关系。若令

$$R^* = I_A$$

$$R^{k+1} = R^k \circ R \quad k = 0, 1, 2, \dots$$

则显然有以下指数定律:

$$(R^{-1})^n = (R^n)^{-1},$$

$$R^n \circ R^m = R^{n+m}$$

$$(R^n)^m = R^{nm} \quad n, m \in \mathbf{N}$$

下面介绍几种重要的二元关系。

相容关系 若集合 A 上的二元关系 R 是自反的和对称的, 则称 R 为 A 上的相容关系。

等价关系 若集合 A 上的相容关系 R 是传递的, 则称 R 为 A 上的等价关系。常用 “ \approx ” 表示。

偏序关系 若集合 A 上的二元关系 R 是自反的、反对称的和传递的, 则称 R 为 A 的一个偏序, 偏序常用 “ \leq ” 表示, 这时称 $\langle A, \leq \rangle$ 为偏序集。

设 $\langle A_1, \leq_1 \rangle$ 和 $\langle A_2, \leq_2 \rangle$ 都是偏序集。如果映射 $f: A_1 \rightarrow A_2$ 满足:

$$\text{若 } a, a' \in A_1 \text{ 且 } a \leq_1 a', \text{ 则 } f(a) \leq_2 f(a').$$

则称 f 为保序映射。如果保序映射 f 为双射, 则称 f 为序同构。

设 $\langle A, \leq \rangle$ 为偏序集 且 $S \subseteq A$ 。

(1) 设 $a \in A$ 。若对每个 $x \in S$ 皆有 $a \leq x$ (或 $x \leq a$), 则称 a 为 S 的一个下界 (或上界)。

(2) 设 $a \in A$ 为 S 的一个下界 (或上界)。若对 S 的每个下界 (或上界) $b \in A$ 皆有 $b \leq a$ (或 $a \leq b$), 则称 a 为 S 的一个下确界 (或上确界)。

显然, 下确界为下界中的“最大者”, 上确界为上界中的“最小者”。因此, 若 S 的下确界 (或上确界) 存在, 则必是惟一的。

(3) 设 $a \in S$ 。若当 $x \in S$ 且 $a \leq x$ (或 $x \leq a$) 时恒有 $x = a$, 则称 a 为 S 的一个极大元 (或极小元)。

(4) 设 $a \in S$ 。若对每个 $x \in S$ 皆有 $x \leq a$ (或 $a \leq x$), 则称 a 为 S 的最大元 (或最小元)。

显然, 若 S 有最大元 (或最小元), 则必是惟一的。

如果集合 A 上的偏序 \leq 满足:

(1) 若 $x, y \in A$, 则必有 $x \leq y$ 或 $y \leq x$;

(2) 若 S 为 A 的非空子集, 则 S 必有最小元。

则称 \leq 为 A 上一个良序, 并称 $\langle A, \leq \rangle$ 为一个良序集。集合 A 上的偏序 \leq 若仅满足上面的条件 (1), 则称 \leq 为 A 上的一个线性序, 这时称 $\langle A, \leq \rangle$ 为一个线性序集。

(王兵山)

guanxi daishu

关系代数 (relational algebra) 研究关系及其运算的代数理论。

n 元关系是由某些 n 元序偶构成的集合。 n 元关系的并、差运算的含义与通常的集合论相同。 m 元关系 R 和 n 元关系 S 的笛卡儿积是形如 $\langle x_1, \dots, x_m, y_1, \dots, y_n \rangle$ 的序偶构成的集合, 其中 $\langle x_1, \dots, x_m \rangle \in R, \langle y_1, \dots, y_n \rangle \in S$ 。 n 元关系 R 在分量 i_1, \dots, i_k ($1 \leq i_1, \dots, i_k \leq n$) 上的投影是形如 $\langle x_{i_1}, \dots, x_{i_k} \rangle$ 的序偶构成的集合, 其中 $\langle x_1, \dots, x_{j_1}, \dots, x_{j_k}, \dots, x_n \rangle \in R, (j_1, \dots, j_k)$ 是 (i_1, \dots, i_k) 的按序排列。关系可视为二维表格, 因此可以给关系的每个列命名。给定关系 R 以及由常量、关系列名、算术比较运算符和逻辑联结词构成的公式 W , R 相对于 W 的选择是指 R 中所有满足 W 的序偶组成的集合。

E. F. Codd 已经证明: 关系代数和关系演算在表达能力上等价。因此, 关系代数和关系演算均可作为评价关系数据库系统查询语言的标准。

基于上述原始的关系算子, 可以定义关系的交、商和连接运算。这些派生算子表达能力强, 并且直观自然, 因此它们在查询语言中得到了广泛应用。

利用关系算子的性质, 可以构造各种算法, 对关系代数表达式进行优化, 以提高关系查询的执行效率。由于关系代数表达式和关系演算表达式之间可

相互转换,因此,这种算法不仅适用于基于关系代数的查询语言,也适用于基于关系演算或两者混合型的查询语言。

关系代数与关系演算共同构成关系数据库的主要理论基础。它在关系数据库的设计理论、查询语言的设计与查询优化等领域有重要应用。例如,目前广为使用的查询语言 SQL 就兼具关系代数和关系演算风格。

参考文献

Ullman J. D. Principles of Database Systems, 2nd ed. Computer Science Press, 1982 (谭庆平)

guanxi shujuk

关系数据库 (relational database) 采用关系模型的数据库。

关系模型用二维表结构来表示各类实体及其间的联系,二维表由行和列组成。一个关系数据库由多张二维表组成。

图 1 是一个关系数据库中中学生人事表的例子:

学号	姓名	性别	系别	年龄	籍贯
9001001	王明	男	计算机	22	江苏无锡
9001002	周晓	女	计算机	22	上海
9001032	张玮	女	计算机	22	北京
⋮	⋮	⋮	⋮	⋮	⋮
9003020	李义山	男	化学	20	山东烟台

图 1 学生人事表

上面的学生人事表是一个六元关系。表中的每一行为一个元组(如 9001032,张玮,女,计算机,22,北京);表中的每一列为一个属性,上表有 6 列,对应 6 个属性(学号,姓名,性别,系别,年龄和籍贯);属性的取值范围称为域,如上表中学生年龄这个属性的域是(14~35),性别的域是(男,女),系别的域是一个学校所有系名的集合;上表中每个学生的学号都不相同,它可以惟一确定一个学生,是本关系的键码;元组中的每一个属性值为分量,如上述元组中的张玮、北京等;关系名(属性 1,属性 2,⋯,属性 n)为关系模式,是对关系的描述,学生(学号,姓名,性别,系别,年龄,籍贯)为此关系的模式。

关系模型是建立在严格的数学概念基础上的。给定一组域(域是值的集合) D_1, D_2, \dots, D_n (这组域中可以有相同域),则其笛卡儿乘积 $D_1 \times D_2 \times \dots \times$

D_n 的子集可以构成一张二维表,称为一个关系。 n 为关系的目或度。表中各列名必须惟一,称为属性名;惟一确定一个元组的属性组称为键码;一个元组中的某一属性称为一个分量。关系模型中关系的每一个分量,必须是不可分的数据项。

关系模式是一个五元组 $\langle R, U, D, DOM, F \rangle$,用于描述关系。其中, R 为关系名; U 为属性名; D 为一组域; DOM 为属性到域的映射; F 为一组数据依赖,是一类完整性约束条件。某一时刻一个关系模式的实例称为关系状态,简称关系。

关系模型的操作部分具有关系处理能力。它把整个关系作为操作对象。具有关系处理能力的关系数据语言,可分为关系代数、关系演算和介于两者之间的语言。一般以关系代数作为度量语言处理功能的标准。关系代数除提供传统的集合运算如并、交、差运算外,还提供了选取、投影、联结等操作。如果由关系代数的选取、投影、联结操作表达的任何查询均能为某种语言表达,而不必使用迭代、递归命令,则可认为该语言具有关系处理能力。对关系数据库进行操作通常采用 SQL 语言。

数据依赖和规范化理论是关系数据库研究的重要内容,数据依赖用来描述数据之间的完整性约束条件,是语义范畴的概念;规范化的概念和范式的定义给出了判别模式好坏的准则,使数据库设计有了评价模式的理论依据,模式分解的概念和算法,又为数据库设计提供了辅助工具。

关系模型是于 1970 年由美国 IBM 公司 San Jose 研究室的研究员 E. F. Codd 发表的题为“大型共享数据库数据的关系模型”论文中首次提出的,开创了数据库关系方法和关系数据理论的研究,为关系数据库技术奠定了理论基础。由于 E. F. Codd 的杰出贡献,他于 1981 年获得 ACM 图灵奖。

20 世纪 70 年代是关系数据库理论研究和开发原型时代。其中以美国 IBM San Jose 实验室开发的 System R 和加州 Berkeley 大学研制的 INGRES 为典型代表。经过大量的高层次的研究和开发取得了一系列的成果,关系数据库从实验室走向了社会。

80 年代以来,计算机厂商新推出的数据库管理系统几乎都支持关系模型,非关系系统的产品也都加上了关系接口。

大量商用关系数据库管理系统的运行,特别是微型计算机关系数据库管理系统的使用,使数据库技术广泛应用到企业管理、情报检索、辅助决策等各个方面,成为实现和优化信息系统和应用系统的基

本技术。

参考文献

1. 萨师煊,王珊. 数据库系统概论. 第二版. 北京: 高等教育出版社, 1991
2. Date C J. An Introduction to Database System. 6th Edition. Reading: Addison Wesley Publishing Company, 1995
3. Ullman J D. Principles of Database and Knowledge-Base Systems. Vol. I, II, Computer Science Press, 1989

(王珊)

guanli guocheng

管理过程 (management process) 软件生存周期中管理者所负责的一系列活动。管理者负责对所从事的过程,例如,对获取、供应、开发和支持等过程的活动进行管理;软件管理过程适用于必须对各自的过程进行管理的任何一方。

软件管理过程的目的是在一定的时间和预算范围内,有效地利用人力、资源、技术和工具,完成预定的系统或软件产品,实现预定的功能和其他质量目标。管理技能往往是系统或软件产品成败和软件质量高低的重要条件。大型软件项目涉及较多的人力、资源和时间,管理问题尤为突出。

软件生产是一项劳动和智力密集的活动,它是以为人中心的过程,具有可见性差和定量化难的特点。可见性差是指软件研制进度不易标志,存在的问题不易及时发现和纠正,其过程容易出现修改和反复。定量化难指软件的成本、生产率和质量不易度量。因此,软件管理有特殊的复杂性。

软件管理过程是随着软件工程的发展而发展的。早在 20 世纪 60 年代末,已经有人讨论大型软件研制项目的组织管理问题。软件工程实践中发生的种种问题,往往是与管理密切相关的,例如,进度推迟、经费超支、质量差、程序人员不称职等。可管理性成为软件生产工程化的重要标志。因此,与软件工程化、产品化过程相适应,形成新的分工,出现了组织管理软件生产的管理员。这些管理员的管理活动体现了最初的软件管理过程。

软件管理活动的研究与软件管理的研究密切相关。最初主要在人员和组织方面,如软件心理学的研究。随着软件技术的发展,软件管理自身出现了专门的方法、技术和工具来支持软件管理活动。

软件管理的对象是进度、系统规模和工作量估计、经费、组织和人员、风险、质量、作业、环境配置

等。因此,软件管理一般可分为进度管理、成本管理、质量管理、人员管理、资源管理和标准化管理等。这些管理一般都有其各自的活动内容,软件管理过程把这些活动归纳起来,抽取活动共性,一般可包含下述活动:

(1) 实施本过程的准备 一开始先要搞清楚进行管理的过程的需求,管理者通过调查研究确认达到需求的可能性。在必要时可通过有关各方协商来修订和完善需求。

(2) 管理计划的制定 制定计划的任务包括规定进度、分配资源、决定与项目有关的组织和承担人员(包括人员的地位、作用、职责、规章制度等)、根据规模和工作量估计分配任务、风险定量化、制定质量管理指标、编制预算和成本、准备环境和基础设施等。

(3) 计划的实施和控制 管理者根据需求对过程进行控制。他们监督过程的实施,提供过程进展的内部报告和按合同规定向获取方提供外部报告,并应当调查、分析和解决在执行过程中发现的问题,对计划进行调整和修改。问题及其解决办法都应写成文档。

(4) 对计划完成程度的评审和评价 管理人员应对计划完成程度进行评审,对项目进行评价,并对计划和项目进行检查,使计划和项目在完成或变更之后保持完整性和一致性。

(5) 管理过程完成时编写文档 管理者根据合同决定此过程是否完成。如已完成,应从完整性方面检查项目完成的结果和记录,并把这些结果和记录编写成文档并存档。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes — IEEE Std 1074 ~ 1991
2. ISO / IEC 12207:1995 Information Technology — Software — Part 1: Software Life-Cycle Process

(刘光龙)

guanli xinxi ku

管理信息库 (management information base, MIB) 网络管理系统中一系列被管对象组。被管对象组是性质相近的被管对象的集合。在网络管理系统中(参见**网络管理**),被管资源的代理通过网络管理协议传送过来的协议数据单元(参见**公共管理信息协议**),接收由管理者发送的管理操作命令,将它作用于管理信息库中的管理对象,再最终映射

到实际的被管资源。管理信息库中包括有被管对象信息的定义和信息本身。

Internet 中的 MIB 采用了与域名系统(DNS)相似的树形结构。最早的管理信息库包含有 8 个管理信息类别,每个类别下面又有若干管理对象,总共约 114 个对象。该管理信息库最初是在征求意见文献 RFC 1066 中发布的,称为“基于 TCP/IP”Internet 的网络管理的管理信息库,即 MIB-I。这 8 个管理信息类别是:系统对象类、接口对象类、地址转换对象类、IP 对象类、TCP 对象类、ICMP 对象类、UDP 对象类和 EGP 对象类。其中,IP、TCP、ICMP、UDP 和 EGP 都属于 Internet 中广泛使用的 TCP/IP 协议集。1991 年发布的 RFC 1213 中又新增加了三个类别,成为 MIB-II。新增加的是 SNMP、CMOT 和传输对象类。

(高传善)

guanli xinxi xitong

管理信息系统 (management information system, MIS) 利用人工过程、数学模型以及数据库等资源为企业、事业单位的各种管理职能提供信息支持的一种计算机应用系统。它是管理人员的有效工具。

管理信息系统对企业、事业单位的作用在于加快信息的采集、传送及处理速度,及时地为各种职能的各级管理人员提供所需的信息,从而改善单位的运行效率及效果。

管理信息系统这个术语虽然早在 20 世纪 50 年代后期就出现了,但当时计算机才开始引入管理,主要用来做例行的事务数据处理,如生产作业统计、会计对账等,目的在于提高事务处理的效率,减轻人的工作量。随着计算机技术的进步,可将关联紧密的多项事务处理功能组织在一个系统中,使数据的交换更直接。到 20 世纪 60 年代后期,由于数据库技术的实用化,覆盖多个职能部门的综合数据处理系统得到迅速发展。开始时,这类系统大多不具备支持决策的能力,一般以提高事务处理的效率为主要目标。随后在综合数据处理系统的基础上逐步增加了分析、计划及控制等功能,支持决策过程。这类系统除了提高事务处理的效率外,还可改善决策,提高经营管理的效果,因而它们不同于以往的数据处理系统,人们便称之为管理信息系统。20 世纪 70 年代出现的决策支持系统可以看作是管理信息系统的发展和延伸。管理信息系统中的辅助决策功能主要

针对运行层及管理控制层的决策过程,而决策支持系统的主要目标在于提高高层的战略级决策过程的有效性。

管理信息系统包括以下基本内容:

(1) 管理模式 一个单位的管理信息系统总是体现或支持一种管理模式的。随着新的管理理念的出现,管理信息系统的内涵也在不断变化。如作为制造业管理信息系统(MIS)重要组成部分的制造信息系统,经历了基于订货点法的库存管理系统、支持物料需求计划的 MRP 系统、支持对所有制造资源进行统一计划和控制的 MRP II 系统的发展历程,到 20 世纪 90 年代,在融合了多种先进管理思想的基础上,出现了对整个企业资源进行统一计划和管理的**企业资源规则系统(ERP)**。

(2) 组成结构 管理信息系统主要是由计算机支撑系统和应用软件系统组成。支撑系统由**计算机网络、数据库系统**及中间件(参见**网络中间件**)等组成,为应用软件系统提供运行环境。支撑系统有两种典型的结构形式,即集中式结构和分布式结构。管理信息系统的功能主要是由应用软件系统实现的。应用软件系统的结构应与单位的管理模式和职能结构相适应,既可支持各种管理职能,也能支持每种职能不同层次上的管理活动。执行每一种职能都需要一组特定的数据和处理功能,它们便形成了管理信息系统(MIS)中各个相对独立的子系统。各子系统之间,借助通信网络、数据库或中间件实现互联及数据共享,使整个系统集成成为一个整体。每种职能的管理活动一般分为 3 个层次:运行层、管理层及战略规划层,管理信息系统的每个子系统均有相应的功能支持这些层次上的管理活动。最底层的功能涉及的数据量最大,且处理过程是预先确定的,结构化、程序化程度最高,越往上,加工处理的数据越综合,数据量越少,结构化、程序化程度越低。一般的管理信息系统对战略规划层的活动支持较弱,这部分功能将由专门的决策支持系统提供。反映以上概念的 MIS 结构如图 1 所示(这里以制造企业的管理信息系统为例)。

(3) 系统开发 MIS 系统的开发主要是应用软件的开发,其开发过程参见**软件工程**。

(4) 系统运行 经验证后的系统可正式投入运行。这个阶段的主要活动有系统的日常运行管理,系统行为的合法性审查,系统功能与性能的监测和评估以及根据审查或评测结果对系统进行维护。

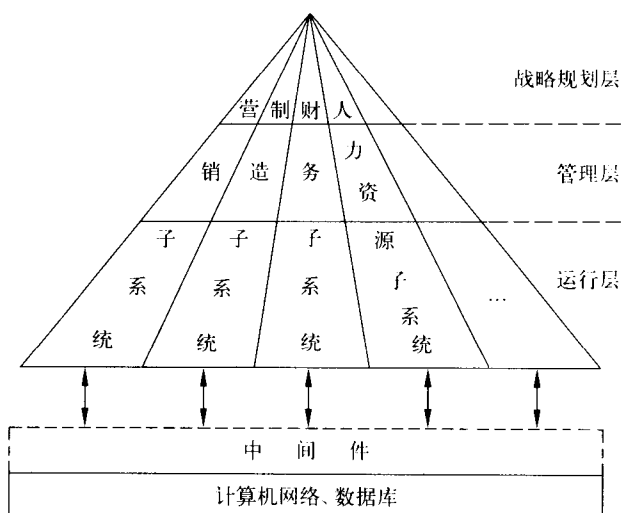


图1 MIS的概念结构

管理信息系统涉及的学科主要有管理学、运筹学、计算机科学及通信学等。

在管理信息系统的开发与运行中,主要问题是计算机系统如何能正确、及时地反映组织的状态及需求。解决此问题有待于组织管理水平的提高及计算机技术的进步。管理信息系统的发展趋势有以下几个方面:

(1) 集成化 管理信息系统将与企业内、外部的其他计算机应用系统互联。如与企业内的办公信息系统、计算机辅助设计(CAD)和计算机辅助制造(CAM)系统相集成。与企业外部互联的系统可能有行业的信息系统、地方和国家的经济信息系统、金融系统以及社会服务信息系统等。

(2) 自然化 人与计算机系统可以用类自然语言进行交互,交互的媒体除文本外,还可以是图形、

图像或声音等,更直观、自然。

(3) 组件化 MIS的软件采用组件结构,组件可以重用,可以组装,能方便地对特定用户进行定制,也能灵活地响应用户多变的需求。

参考文献

Raymond McLeod Jr., George Schell. 管理信息系统, 管理导向的理论与实践. 张成洪等译. 北京: 电子工业出版社, 2002 (董逸生)

guanli zixitong

管理子系统 (administration subsystem) 结构化布线系统中对布线电缆进行端接及配线管理的子系统。管理子系统所处的位置如图1所示。

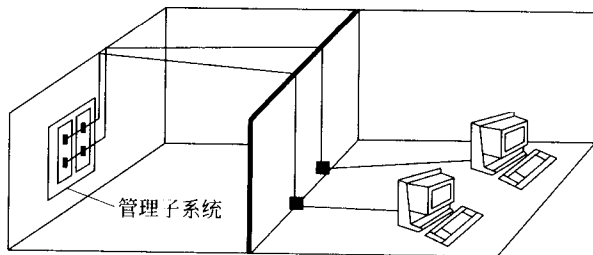


图1 管理子系统

管理子系统通常设置在一幢大楼的中央设备机房和各个楼层的分配线间。一般由配线架和相应的

跳线组成。管理子系统所在的房间称为管理间。

通过管理子系统,用户可以在配线架上灵活地

更改、增加、转换、扩展线路,而不需要专门工具或借助专业技术人员。正是由于这些功能,结构化布线系统才具有传统布线无法比拟的开放性、扩展性和灵活性。

管理间的设计及建设应考虑这样一些因素:

(1) 位置 各楼层的管理间一般设置在弱电竖井内,由于双绞线用于计算机网络数据通信时受距离限制,所以管理间的设置要首先尽量满足传输距离的要求。

(2) 大小 应根据管理间内端接的信息点的数量、采用的配线架的大小和数量以及要放置的互连设备的大小和数量确定管理间的大小,还要注意留有足够的操作空间。

(3) 环境要求 管理间要注意防尘、防火,并且要根据设备的要求及有关规范保证一定的温度和湿度。

(4) 标识 为了管理维护的方便,在管理间内,管理子系统端接的线缆都应有明确标识。同时对于管理间内的配线架也应做明显的标志。根据配线架所端接的线缆功能划分不同的区域,使用不同的颜色来区分。

(5) 配线架排列 对管理间内的配线架的排列应遵循这样一些原则:

①进出线方便;②跳线方便,尽量短;③墙面布局合理,占用空间小。

结构化布线系统管理了系统中的光缆部分简称为光缆管理子系统。光缆管理子系统的主要元件有:光缆接线盒、过线槽、光纤跳线、接头(ST、SC等)、各类光纤耦合器。

光缆管理子系统是实现主干光缆互连、交连的场所,通过它可以实现光纤与光纤之间、光纤与设备之间的灵活跳接。光缆管理子系统的标识对于系统的维护和管理也非常重要,一般光缆管理子系统的标识应包括以下一些信息:

- ① 光缆编号;
- ② 光缆远端连接的位置;
- ③ 光缆长度;
- ④ 光缆芯数,指明使用的光纤数及备用的光纤数;
- ⑤ 光缆的类型,指明是单模光缆还是多模光缆。

参考文献

胡道元主编. 网络设计师教程. 北京: 清华大学出版社, 2001

(王晓东)

guangbi

光笔 (light pen) 一种外形像笔, 上有按钮, 以缆线或无线方式和主机相连, 与显示器配合使用的输入设备。使用者把笔指向屏幕上某一点, 按动按钮, 可以进行在屏幕上作图、改图、图形放大、移位等操作。

光笔的结构如图 1 所示。笔端有一小孔, 用来接收荧光屏上发出的光。通常在屏幕上有一个十字形光标供光笔使用。当需要对光标所在位置的图形进行操作时, 把光笔对准光标, 并按动按钮。光笔接收到的光经光电变换、放大、整形后送入计算机。由于显示器采用光栅扫描方式, 因而通过扫描电路可以判断出光笔所指点的位置的 x, y 坐标, 由软件作进一步的操作处理。

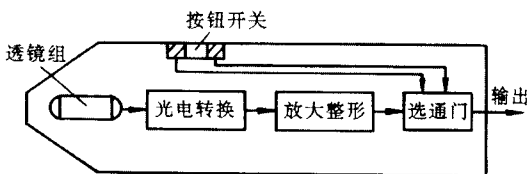


图 1 光笔结构

光笔有两种使用方式:

(1) 指点方式 光笔选择图形或字符, 进行删除、修改、旋转、放大等操作。

(2) 作图方式 光笔对准光标在屏幕上以一定速度拖动, 使光标在屏幕上画出所需要的曲线, 用来画布线图、机械图、写字等。

光笔的使用功能依赖于硬件和软件。软件越丰富, 光笔能实现的功能也越多。

光笔价廉, 使用方法简单, 使用者不必了解复杂的控制程序。特别是用于作图时, 非常方便。但长时间使用, 举笔的手非常容易疲劳, 这个缺点使它的使用受到了限制。

(林兼)

guang cunchuqi

光存储器 (optical storage) 用光学方法从光存储媒体上读取和存储数据的一种设备。对光存储媒体最基本的要求是, 存储单元的光学性质可以用物理方法改变以代表被存储的数据, 同时这种性质改变可以用光的方法检测 (读取) 出来。目前几乎所有的光存储器都使用半导体激光器作为光源, 因而光存储器也称为激光存储器。

光存储主要包括激光逐位存储、全息存储和其

他光存储。广义来说,光存储器还应包括**条码阅读器**、光电阅读机等。但在计算机领域,光存储器一般指光盘机、全息存储器、光带机和光卡机等设备,其中光盘机应用最广。光盘机主要有**只读光盘驱动器**(CD-ROM 和 DVD-ROM)及可改写光盘驱动器(CD-R, CD-RW, DVD-RAM, MO 和 PD)等多种类型。另外,全息存储系统也已实用化,产品有专用的全息存储系统(例如用于空间技术)和采用全息盘片作存储媒体的存储器。

光盘机主要由数据读写头、伺服机构和主轴系统、编解码和纠错系统、数据通道等部分组成,光盘机的主要技术指标是数据传输速率、寻道时间、可支持的最大容量、可支持的数据格式、记录的误码率和出错几率、可支持的接口标准。

光存储器技术发展很快,由 CD 驱动器发展到 DVD 驱动器。蓝光光盘驱动器技术已经成熟,可将存储密度提高到 20 GB/in²。

参考文献

1. Bradley A C. Optical Storage for Computers Technology and Applications. New York: John Wiley & Sons, 1989
2. 张守仁. 光盘存储器. 北京: 科学出版社, 1989 (裴先登)

guangdian jichengdianlu

光电集成电路 (optoelectronic integrated circuit, OEIC) 完成光信息与电信息转换的一种集成电路。光电集成电路有两类:一类是完成光信息到电信息转换的电路,它由光电探测器、放大器及偏置电路组成;另一类是完成电信息到光信息转换的电路,由光发射器件、驱动电路及偏置电路组成。常见的发射器件有发光管、激光管、液晶,而接收器件有光电晶体管、硅光电池等。

光电集成电路根据其处理的光信息不同可分为红外光、可见光及激光三类。

光电集成电路已广泛用于照相机、电视摄像、工业自动控制、传真和光纤通信,以及机器人与视觉传感器、平面显示、夜视、卫星通信和导航等国民经济各个领域。近年来,计算机互联网络及电话光交换的研究,进一步扩大了它的应用范围,并促进光电集成电路从早期完成单路光电信息转换向多路及面阵方向发展。

为了实现图像的光电转换,在光电集成电路中必须集成调制器、多路复用及控制电路,从而使得现

今的光电集成电路包含大规模的信息传输和处理电路。此外,还引进了各种新的器件以提高光电集成电路的性能价格比。例如,在红外与电视的摄像机中都采用了电荷耦合器件。

新器件的引进使光电集成的工艺比常规的集成电路工艺更为复杂化。目前比较成熟的大规模集成电路是用平面工艺在硅基体上研制而成,其主要有源器件有双极晶体管、MOS 场效应晶体管及肖特基场效应晶体管三种,从而形成三类硅平面集成电路。但由于硅的禁带宽度(1.12 eV)只适合于制备可见光波段的探测器,而红外探测器必须在禁带宽度窄的化合物半导体上制备,激光器又必须在砷化镓衬底上制备,因此如何将三类器件晶片统一起来是使光电集成电路从过去适合于光接收器的小规模光电转换电路,或适合于光发射器的小规模光电转换电路,发展到大规模光电集成电路所必须解决的关键问题。

(林雨)

guang jisuanji

光计算机 (optical computer) 主要利用光技术和光器件实现的计算机。同电子技术相比,光技术有如下显著的特点:①传输速度快;②互连数大,互连密度高,一般光学系统的互连数可达 10⁶ 个光点数;③非物理触点互连,这大大提高了可靠性和互连密度,同时,互连空间可以重复使用;④遵循独立传输原理,上亿道光信息可以汇聚在一起,按照各自的目的地,独立、无干扰地传递信息;⑤光的载波空间带宽可达 100 THz(电信号带宽最多为 100 GHz),而且信息传输无失真;⑥功耗低,光信号衰减小。

光计算机可分为两大类:模拟光计算机和数字光计算机。1960 年第一台激光器的产生提供了空间和时间干涉性极好的光源,使得光学模拟计算机的研究走向了高潮。但由于模拟量计算远不如数字逻辑运算的精度高,实用性有限。1979 年第一个半导体光学双稳器件研制成功。1983 年,提出了研究数字光计算机的构想。此后,全光数字计算机的研究进入高潮。

虽然 1990 年宣布了第一台全光数字计算机研制成功,但由于工艺上的困难,光器件的性能、功耗、体积以及互连数等方面均不及电子器件,所以后来没有继续再向全光数字计算机方向发展,而倾向于采用光电混合的方法来研制光电混合型计算机。在这种计算机中,把硅和砷化镓集成技术用于实现传

统的数字电路,而芯片、部件等间的互连则由光器件来实现。发展这种光计算机的关键在于研制出体积小、能耗少、成本低且易于制造的光电子转换器,主要是微型激光二极管和光电二极管,同时要研制能精确控制光路的制动定位系统。

参考文献

Miles J Murdoca. A Digital Design Methodology for Optical Computing. Cambridge, MA: MIT Press, 1990
(刘德才)

guang jilu

光记录(optical recording) 用光学方法在光存储媒体上记录和读取数据的技术。光记录的基本原理是:光存储媒体上存储单元的光学性质可以用物理方法改变以代表被存储的数据,同时这种性质的改变可以用光学方法检测(读取)出来。

数据在媒体上的录入(记录)方法用得最多的是光的热效应方法和机械方法。用光的方法读取就是对存储单元的光学性质(如反射率、偏振方向等)进行辨别,并转化为便于检测的电信号。光存储用的光电传感器大都是检测光强的。由于具体实现上的原因,几乎所有的光存储器都检测反射光而不是透射光。最常用的方法是利用存储单元不同的反射率来表示存储的信号是1或0。另一种用得较广的方法是用反射光的偏振方向来表示存储的信号,这时就要用光学的方法将偏振方向转化为落在光电传感器上的光强。

光记录技术是一个光、机、电结合的技术,主要涉及光学系统(信息的写入和读出)、精密机械(伺服系统)、编解码和纠错技术、数据通道技术等。

光记录可分成只读光记录和可改写光记录两种类型。只读光记录的记录媒体是用机械或热致形变方法产生的,如CD-ROM光盘和DVD-ROM光盘是用模压生产的;可改写光记录的记录媒体有染料化合物、相变材料、磁光材料和全息材料,其中用染料化合物生产的CD-R光盘是一写多读媒体,相变材料制造的CD-RW光盘和磁光材料制造的MO光盘是可重写光盘。2003年,CD-RW光盘可重写1000次以上,MO光盘可重写100万次以上。全息材料用于全息存储,具有立体存储和高速并行的特点。

光记录技术具有下述特点:

- (1) 易达到高密度记录;
- (2) 比较容易实现记录媒体的更换,而且记录媒体一般比较廉价;

(3) 对纠错措施要求较高。

光记录技术还处于快速发展阶段,高密度光记录技术有短波长记录、多层多阶记录、磁区放大读出技术、超分辨率读出技术等。

参考文献

1. 徐端颐. 光盘存储系统设计原理. 北京:国防工业出版社,2000
2. 干福熹等. 数字光盘存储技术. 北京:科学技术出版社,1998 (谢长生 黄浩)

guang jilu bianma fangfa

光记录编码方法(encoding method of optical recording)

按照数字光记录的需要为提高记录密度和可靠性而采取的代码变换。其基本理论、编码方法和工程实现等与数字磁记录编码方法相似。在光记录中采用的编码有:8/14调制码(EFM)、4/15码、RLL(2,7)码、RLL(1,7)码等。

EFM码 是一种无直流分量码。它将8位数据码转换成14位通道码。由于 2^{14} 有16384种二进制码, 2^8 只有256种,因此需从16384种选择256种。选择的要求是:两个1之间最少要有2个0,最多有10个0。这种选择方法得到的转换表可在标准ISO/IEC 10149中查到。表1列出了部分编码作为示例。EFM码也是一种游程长度受限码,它的参数 $d=2, k=10$ 。为了消除直流分量,在14位后加3位,这3位根据情况在000,001,010,100中选择一个。因此实际编码是17位。

表1 EFM码变换表(部分)

序号	数据	通道码
0	00000000	01001000100000
1	00000001	10000100000000
2	00000010	10010000100000
4	00000100	01000100000000
32	00100000	00000000100000
63	00111111	00100000001000
127	01111111	01100000000010
255	11111111	00100000010010

4/15码 是光盘B格式(离散采样伺服格式)中采用的一种记录码型。它无自同步能力,读出的时钟信号必须从光盘片预格式化的时钟信号中导出。4/15码是将数据的一个字节分成高4位和低4位分别处理变换,即将8位数据变换为15位通道码,表2为其部分编码。编码的要求是,两个1之间

0 的个数最少为 0, 最多为 10, 每个编码都有 4 个 1。

表 2 4/15 码编码表(部分)

数 据		通道码
二进制	十六进制	二进制
00000000	00	1100110000000000
00000001	01	1100011000000000
00011110	1E	0100000110001100
00011111	1F	1110000100000000
11111110	FE	0000000010011110
11111111	FF	000000011100100

RLL(2,7) 码 广泛用于当今温式磁盘机及高档数据流磁带机中, 也为盘径 130 mm 的磁光盘所采用。其约束参数 $d=2, k=7$, 表示记录序列中相邻两个 1 之间至少有 2 个 0, 连续 0 的个数最多为 7。其编码规则见表 3。

表 3 RLL(2,7) 码编码规则

数据序列	记录序列
10	0100
010	100100
0010	00100100
11	1000
011	001000
0011	00001000
000	000100

RLL(1,7) 码 已在新的磁光盘驱动器中获得应用, 130 mm 盘径 2 GB 容量的磁光盘以及 90 mm 盘径 640 MB 容量的磁光盘记录码型均采用 RLL(1,7), 其编、译规则分别如表 4、表 5 所示。

表 4 RLL(1,7) 码编码规则

前一位记录位 e_0	当前待编码数据	后继输入数据	记录序列 $e_1 e_2 e_3$
$\times^{①}$	00	0 \times	001
0	00	1 \times	000
1	00	1 \times	010
0	01	0 \times	001
0	01	1 \times	000
1	01	00	010
1	01	not 00 ^②	000
0	10	0 \times	101
0	10	1 \times	010
0	11	00	010
0	11	not 00	100

注: ① \times 表示该位取值为 1 或 0;

② not 00 表示可以是 01, 10, 11 三种序列中任意一种。

表 5 RLL(1,7) 码译码规则

前二位记录序列	当前待译码记录序列	后继记录序列	译码后数据位 $d_1 d_2$
10	000	$\times^{①} \times$	00
not 10 ^②	000	$\times \times$	01
00	001	$\times \times$	01
not 10	001	$\times \times$	00
$\times 0$	010	00	11
$\times 0$	010	not 00 ^③	10
$\times 1$	010	00	01
$\times 1$	010	not 00	00
$\times \times$	100	$\times \times$	11
$\times \times$	101	$\times \times$	10

注: ① \times 表示取值可为 1 或 0;

② not 10 表示可以为 00 或 01 或 11 序列;

③ not 00 表示可以为 01 或 10 或 11 序列。

关于 RLL(2,7) 和 RLL(1,7) 码的性能可以通过它们的结构参数 $(d, k; m, n, r)$, 即 $(2, 7; 1, 2, 1)$ 和 $(1, 7; 2, 3, 1)$, 根据游程长度受限码(RLLC)的理论计算出来。

在光盘存储设备中, CD-ROM 只读型光盘机曾采用过 EFM 码, 磁光型可改写光盘机 A 格式中采用 RLL(2,7) 码, B 格式中采用 4/15 码。它们均为二元(0 或 1)编码, 编码效率 $M = (d+1)m/n \leq 1.5$, 其中 m 是数据序列长度, n 为编码后记录序列长度(位数), d 为两个 1 之间 0 的最少个数。目前二元编码的效率不超过 150%。G. V. Jacoby 提出的三元 3PM 码的编码效率 $M = 2$, 检读窗宽 $T_w = 2/3$, 优于二元 3PM 码和 RLL(2,7) 码。三元编码是记录编码的发展趋势之一。

光记录系统中的检纠错编码和数字磁记录系统中的纠错码(ECC)相似。1989 年 10 月公布的 ISO/IEC-10089 标准文件中给出光盘 PEP 区所用的循环冗余检验码(CRC)的生成多项式为 $G(x) = x^8 + x^4 + x^3 + x^2 + 1$; 在 SFP 区的标志(ID)部分, 给出 CRC 的生成多项式为 $G(x) = x^{16} + x^{12} + x^5 + 1$ 。

参考文献

Jacoby G V. Ternary 3PM Magnetic Recording Code and System. IEEE Trans. Magnetics, 1981, MAG-17 (6)

(刘庭华)

guanglan

光缆(fiber optical cable) 利用光而非电信号

来传输信息的一种传输介质。光缆通信具有传输距离远、损耗低、频带宽、串扰小及抗电磁干扰能力强等特点。与双绞线电缆及同轴电缆一样,光缆既可以用来传输模拟信号,又可以用来传输数字信号。

光缆有单模和多模之分,其特性比较如下表所示:

单 模	多 模
用于高速度、长距离传输	用于低速度、短距离传输
成本高	成本低
窄芯线,需要激光器	宽芯线,聚光好
耗散小,高效	耗散大,低效

光缆的类型由所采用的模材料(玻璃或塑料纤维)及芯的尺寸决定,芯尺寸的大小决定光的传输质量。通常用的光缆有以下几种:

- (1) 8.3 μm 芯/125 μm 外层,单模;
- (2) 9.5 μm 芯/125 μm 外层,多模;
- (3) 50 μm 芯/125 μm 外层,多模;
- (4) 100 μm 芯/140 μm 外层,多模。(王晓东)

guangliangdu jisuan

光亮度计算(luminance calculation) 对特定光源在物体表面产生的光照强度所做的计算。该计算仅考虑简单实用的光照模型,物体不透明,无透射光线,因而物体表面呈现的颜色和亮度仅由反射光决定。反射光通常由环境光、漫反射和镜面反射3个分量组合而成,并可用下式表示

$$I = k_a I_a + \sum_i (k_d I_d + k_s I_s)$$

式中, I_a, I_d, I_s 分别表示环境光、漫反射和镜面反射的光照强度。其中后面两项分别要对每个光源(i)进行累加计算。

环境光反射分量是由物体周围的环境光产生的。假定光线均匀地从周围环境入射至景物表面,并等量地向各个方向反射出去,因而环境反射光分量为常量。漫反射分量是光源入射后经物体的粗糙表面向各个方向均匀反射而引起的,其大小与接受方向无关。漫反射计算服从 Lambert 定律:反射光强与光源入射角的余弦成正比,即

$$I_d = I_i \cos \alpha_i$$

式中, I_i 为光源 i 的光强, α_i 为该光源的入射角(是入射方向 L 与法线方向 N 的夹角),见图1。镜面反射是由物体表面光滑特性引起的带有方向性的反射。它遵守光的反射定律:反射光线与入射光线相

对于表面法线对称。使用时,镜面反射分量常使用以下公式来模拟:

$$I_s = I_i \cos^n \beta_i$$

式中, β_i 为反射光线向量 R 与视线向量 V 之间的夹角, n 为正整数(通常取 $2 \leq n \leq 200$)。这样,一个简单实用的光照计算模型可表示为

$$I = k_a I_a + \sum_i [(k_d \cos \alpha_i + k_s \cos^n \beta_i) I_i / (d^2 + C)]$$

式中, k_a, k_d, k_s 分别为环境光反射、漫反射和镜面反射的比例系数,这些系数取决于物体表面的属性; d 为光源至物体的距离; C 为一特定常数。以上计算模型于1973年由 B. T. Phong 提出,因而称为 Phong 模型。

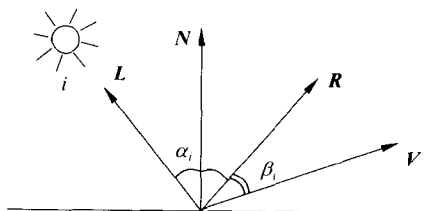


图1 光照计算示意图

参考文献

1. 唐泽圣,周嘉玉,李新友. 计算机图形学基础. 北京:清华大学出版社,1995
2. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2001

(吴思华)

guangpan jingxiang fuwuqi

光盘镜像服务器(CD/DVD mirror server)

将大量 CD 系列或 DVD 系列光盘片的内容完整复制到硬盘,并通过网络来提供光盘内容服务的设备。

使用该设备首先要将光盘的内容完整复制(即镜像)到硬盘。该过程十分简单,将光盘片放到设备中的 CD-ROM 或 DVD-ROM 驱动器中,镜像程序辨识盘片的种类后,自动将光盘的内容复制到硬盘。一张光盘复制完毕后,光盘自动弹出,即可换下一张光盘进行镜像。一旦光盘的内容复制到硬盘,光盘片就不再需要,服务是基于硬盘中的内容来提供的。一个设备能镜像的光盘盘片数取决于光盘片的种类和设备中硬盘空间的大小。对 CD 系列的盘片(650 MB),一般光盘镜像服务器能镜像的盘片数量在 100 片到 1 000 片之间。

光盘镜像服务器属于附网存储设备(NAS),它通过网络接口直接挂接在网络上。由于是网络共享的存储设备,一般要求它支持多种网络文件系统和协议。支持的网络文件系统主要有 Microsoft 平台的 SMB/CIFS, UNIX /Linux 平台的 NFS over UDP/IP, Novell NetWare 平台的 NCP over IPX。支持的网络协议主要有 HTTP over TCP/IP 以及 TCP/IP 组件(RAPP, BOOTP, DHCP, ICMP, WINS 等)。

一般要求光盘镜像服务器支持各种盘片格式,其主要格式有 CD-DA(Audio CD), CD-ROM, CD-R, Video-CD, CD-Extra, Photo CD 以及 DVD 等。

光盘镜像服务器是一种支持光盘资源共享的瘦服务器,支持多个网络用户通过各种文件共享协议进行数据访问。特别地,它还具有低成本提供视频点播的功能,将视频 VCD, SVCD 和 DVD 盘片的内容镜像到硬盘后,对网络客户端提供视频点播服务。光盘镜像服务器支持 MPEG-1, MPEG-2, MPEG-4 和 RM 等视频编码方式。支持同时点播的用户数取决于设备的中央处理器性能、内存大小、硬盘的速度、网络的性能以及文件放置的策略,一般在几十个用户左右。

光盘镜像服务器还具有管理功能,主要有光盘目录管理,盘片种类的识别与分类,用户权限管理,镜像替换策略,内容的搜索与查找,网络设置等。管理程序是通过客户端的浏览器来访问的,进入管理界面而需要有管理员的权限。

光盘镜像服务器使大量的光盘资源成为网上共享资源。与光盘塔和光盘库相比,它一般只需要一个光盘驱动器,不需要机械换盘装置,因而价格低廉;又因为这些内容是复制到硬盘后被用户访问的,因而它的访问速度要比光盘塔和光盘库高得多。

光盘镜像服务器的应用领域十分广泛,主要有图书馆、资料室、档案馆、教育行业、政府机关和中小企业等。

参考文献

1. (美) Marc Farley 著. SAN 存储区域网. 孙功星, 蒋文保, 范勇等译. 北京: 机械工业出版社, 2001
2. 徐端颐. 光盘存储系统设计原理. 北京: 国防工业出版社, 2000 (谢长生)

guangpan kongzhiqi

光盘控制器 (optical disc controller) 连接计算机与光盘驱动器,用于对光盘驱动器进行数据

存取及状态控制的部件。它将计算机需要存储的数据,经变换和处理后,送往能执行写数据的光盘机中由计算机程序指定的地址处进行记录;或从光盘机中取出计算机需要使用的数据,经处理和变换后,送往计算机主存,实现计算机对光盘机的数据存取;并及时检测光盘机的状态及其他信息,提供给计算机处理。

光盘控制器可按光盘机的类型分为三类:只读光盘机控制器、一写多读光盘机控制器和可擦光盘机控制器。它们的控制对象不同,功能、结构和工作原理也有一些差别。只读光盘机控制器只控制只读光盘机完成读数据到计算机主存的有关操作。一写多读光盘机控制器除了具有读功能外,还可根据用户程序的要求在数据段空白的盘片上写入数据,但只能写入一次,以后就和只读光盘控制器一样,对已写入的数据只读不写。空白盘片是预先格式化了的,即事先已录好导向槽及扇区识别段的地址等信息。可擦光盘机控制器则可根据用户程序的要求控制并完成重复读、写及其有关操作。上述三种控制器就功能而言,可以认为前两种是可擦光盘机控制器的特例。各种控制器中实现相同功能的逻辑电路结构和工作原理可有很多相似之处,但由于各种光盘机性能上的差异仍各有特点。

可擦光盘机控制器具有按规定的光盘格式读、写数据的能力,控制器的组成部件及其工作原理和硬磁盘控制器类似。一般有两条信息通路:数据通路和控制通路。对于数据通路,要设置缓冲存储器,其容量可根据数据传输速率、扇区长度、每圈的扇区数、直接存储器存取(DMA)突发传输长度等因素确定。为实现写操作,要有串并转换电路、检纠错码编码电路、调制码编码电路等。光盘不存在磁记录那样的峰点漂移问题,无需写补偿环节。为实现读操作,要有数据分离电路、检纠错码译码电路、串并转换电路等。光盘机的误码率较高,其数据的检纠错码大多采用检纠错能力较强的交叉交错 R-S 码(CIRC)。光盘控制器的控制过程大体上分为三点:①寻找要读或写的轨道;②搜索要读或写的扇区;③读出或写入扇区数据段的数据,对数据进行变换、处理和传送。为此,要计算光学读出头现行位置和目标位置的差值,发出寻道控制信号控制光学头准确寻道和正确聚焦;要将读到的扇区地址与所要求的地址进行比较,找到所需扇区;要形成与读、写过程有关的序列控制信号,使读、写数据在数据通路上传送,完成相应的变换和处理。还需随时根据控制

器采集到的寻道、读、写过程中的各种状态信息,进行相应的控制。以上所需的控制逻辑可用多种方法实现,现在普遍采用单片微处理机或通用微处理机进行控制,使控制器智能化。只读光盘机控制器无需与写功能有关的部分,一写多读光盘机控制器无需写格式的功能。

光盘控制器作为一个接口部件,一端和具有设备级接口的光盘机相连,另一端与主机通道或总线相连。主机侧一般设计若干接口寄存器,如数据寄存器、控制和状态寄存器以及一些逻辑电路。光盘控制程序通过相应端口对这些寄存器寻址,进行读取状态、发送命令和传送数据等操作。控制逻辑根据命令性质和有关参数,形成光盘机接口所要求的控制信号,并控制读写数据完成串并或并串转换以及编译码等处理及必要的缓冲,将读数据送主机,写数据送光盘机。通过光盘机接口传送的控制信号、数据和状态信号是通过发送、接收驱动电路传送的,一般采用菊花链方式与光盘机相连。

为了获得高传输速率、设备独立性、高容量和高智能等优良性能,很多光盘机都采用 SCSI 作为设备的接口,将传统的控制器功能直接集成到光盘机内部,形成嵌入式控制器,通过匹配主机接口与 SCSI 接口的主机适配器将光盘机接向主机。如果主机本身配置 SCSI 总线,则此种光盘机就可以直接与主机相连。

微型计算机的光盘适配器也常称为光盘控制器。

参考文献

1. 张守仁. 光盘存储器. 北京: 科学出版社, 1989
2. 吴产东, 郭学理编著. 微机磁盘光盘系统原理与维修. 武汉: 武汉大学出版社, 1991 (叶济忠)

guangpanku

光盘库 (optical disc library) 一种以光盘为基本存储单元, 可存放多片光盘, 并由机械臂装置选取其中一片在光盘驱动器上装卸并进行存取数据的设备。又称**光盘自动换盘机**。用于光盘库的光盘是普通光盘, 所用的光盘驱动器也是普通的驱动器, 不需要专用或特制的。只读光盘、一写多读光盘、可擦写光盘都可组成光盘库, 这些光盘的盘径有 14 英寸、12 英寸、5.25 英寸、3.5 英寸等多种规格。

光盘库的基本结构包括: ①有许多插槽的光盘架, 用于插放光盘; ②一个或几个光盘驱动器;

③换盘机构, 亦即将光盘从插槽中取出并装入驱动器或从驱动器取出并放回插槽的机械手; ④控制器, 接受来自主机的命令并协调前三部分完成相应的动作。

2003 年光盘库的存储容量一般为 500 GB 到 4 TB 之间。光盘库内的盘片数一般为几十片到数百片, 典型的驱动器数量为两个到六个。换盘时间 (包括盘片运送时间、盘片装卸时间以及主轴达到工作转速的启动时间) 的平均值约为 6 s 左右。平均故障间隔时间为 5 000 ~ 10 000 h。接口主要采用 SCSI 接口和 RS-232 串行接口。

光盘库适用于存储大量不常用但又需要很快就检索到的数据, 如电影、电视节目、专利文献等。

参考文献

- Bradley A C. Optical Storage for Computers Technology and Applications. New York: John Wiley & Sons, 1989 (裴先登)

guangpanta

光盘塔 (optical disc tower) 一种由多个光盘驱动器 (一般为 CD-ROM) 组成的多光盘存储设备。光盘预先放置在各个光盘驱动器中, 用户访问光盘塔时, 可以直接访问各个驱动器中的光盘。根据与服务器接口的不同, 光盘塔通常分为 SCSI 光盘塔和网络光盘塔两种。

SCSI 光盘塔通过 SCSI 总线连接到服务器上, 用户访问光盘必须经过服务器; 网络光盘塔为第二代光盘塔产品, 内置微处理器和系统软件, 可以直接连接到网络, 无需服务器支持即可独立运行, 用户通过 NFS 与 SMB/CIFS 等网络协议访问光盘塔中的光盘。

受 SCSI 总线 ID 号的限制, 光盘塔中的 CD-ROM 驱动器一般以 7 的倍数 (7, 14, 28, 63) 出现。光盘塔中的光盘驱动器可以采用 CD-changer, CD-ROM, DVD-ROM 和 HYPER CD 等几种形式。系统平均寻道时间、持续数据传输速率与光驱的性能指标相关, 平均故障间隔时间为 50 000 ~ 100 000 h。

由于光盘塔通过光驱直接访问光盘, 故速度慢且不支持并发用户同时访问。另外, 由于光盘塔包含多个光盘驱动器, 故设备体积大、价格高, 正逐渐被新的产品如**光盘镜像服务器**等取代。

光盘塔适用于存储大量不常用但又需要快速检索的光盘数据, 如视频数据、专利文献、电子图书资料等。

参考文献

徐端颐. 光盘存储系统设计原理. 北京: 国防工业出版社, 2000
(谢长生 谭志虎 刘瑞芳)

guangxian dao lubian

光纤到路边 (fiber-to-the-curb, FTTC)

采用光纤作为传输介质并接到用户家的路边的宽带接入体系结构。

在 FTTC 结构中, 数字信号从服务提供者经主干链路传到中心局 (CO), 再从 CO 传送到光纤网单元 (ONU)。在 ONU, 光纤信号转换为电信号, 然后通过铜线或同轴电缆传到客户, 有时也可通过无线传输。

电话公司提供的服务使用双绞线从路边连接到客户场地。而有线电视供应商则使用同轴电缆。FTTC 通常由使用多根光纤的交换网络实现, 以传输双向信号流。FTTC 也可同 HFC 体系结构配合使用。参见混合光纤同轴电缆。

FTTC 网络从 ONU 到每个用户的下行信息的传送速率为 51 Mb/s, 上行速率则为 1.62 Mb/s。以 51 Mb/s 的传输速率交换信息流可同时携带 6~7 个高质量的数据流到每个用户。FTTC 能为每个用户提供各种高速数据服务和电话服务, 并可为每个用户提供相当可观的频带。然而, 对 FTTH 这类网络结构目前还没有很大的客户需求, 因为目前使用 VDSL 双绞线结构已能有效地、廉价地提供能满足用户需要的频带。

参考文献

1. 胡道元. 智能建筑计算机网络工程. 北京: 清华大学出版社, 2002

2. Padmanand warrier, Balaji Kumar, XDSL Architecture. McGraw-Hill Inc., 2000 (胡道元)

guangxian fenbu shuju jiekou

光纤分布数据接口 (Fiber distributed data interface, FDDI)

以光纤为传输介质, 以权标传递为介质访问控制方法、数据速率为 100 Mb/s 的一种双环结构高速网络。FDDI 标准由美国国家标准学会 (ANSI) 特许委员会 ASC X3 T9.5 负责制定。该标准与权标环网介质访问控制标准 IEEE 802.5 (参见局域网协议标准) 相似, 但作了一些改变。表 1 总结了 FDDI 与权标环网的主要区别。有些区别在物理层, 有些在介质访问控制 (MAC) 子层 (参见局域网基准 (参考) 模型)。

表 1 FDDI 与权标环网的主要区别

比较内容	FDDI	权标环网
拓扑结构	双环、故障后可重构	单环、不可重构
数据速率	100Mb/s	4/16Mb/s
信号速率	125Mbaud	8/32Mbaud
最大帧尺寸	4500 字节	4500 字节 (4Mb/s) 18000 字节 (16Mb/s)
可靠性要求	有	无
信号编码	4B/5B (光纤) MLT (双绞线)	差分曼彻斯特
传输媒体	光缆、UTP、STP	UTP、STP
时钟	分布式	集中式
容易分配	计时权标轮转	优先级与预定位
权标释放	传输后释放	数据被始发站接收后释放

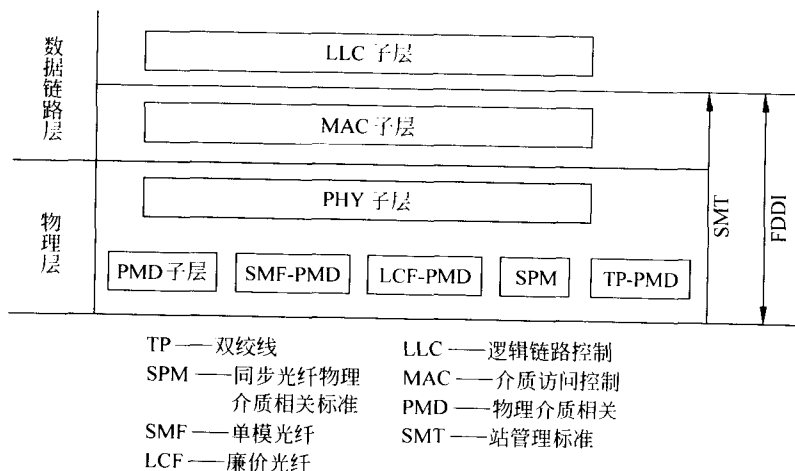


图 1 FDDI 的层次模型结构

FDDI 的层次模型结构分物理层和数据链路层(见图 1)。物理层又分两个子层即**局域网物理介质相关子层(PMD)**和**物理子层(PHY)**。PMD 子层依使用介质类型不同而不同。当用多模光纤时,使用 PMD 标准,当用单模光纤时,使用 SMF-PMD 标准;使用廉价光纤时为 LCF-PMD 标准;当与同步光纤网(SONET)连接时使用 SPM 标准;当使用双绞线时使用 TP-PMD 标准。此时称为铜线分布式数据接口(CDDI)。介质 PHY 子层起到将 PMD 子层与数据链路层连接的作用。数据链路层分为**局域网介质访问控制(MAC)子层**和**局域网逻辑链路控制(LLC)子层**。图中右边的站管理(SMT)标准对三个子层进行协调管理,同时还对 LLC 层及以上各层进行管理。其主要管理功能有:站点的连接与拆除管理、站点初始化、内部配置管理、统计信息搜集和地址管理。

FDDI 网络的拓扑结构与带星形或树形的双环结构。图 2 是 FDDI 拓扑结构的一个例子。网络由站点和光纤传输介质组成。网中有三类站点:①双连站(DAS) 用于建立双环和连接端用户站点;②双连接集中器(DAC) 用于建立双环、单环和连接端用户站点;③单连接站(SAS) 通过单环与 DAC 连接。图中由 DAS 和 DAC 构成双环,其中外环为主环,内环为副环。正常工作时,主环是信息传输的主通道,副环作为备用环。当光纤或站点发生故障时,则启动备用环,利用主环和副环重新构成环路。图 3 显示了 FDDI 网络的重构功能。图 3(a)正常工作时由主环传递信息。图 3(b)和图 3(c)分别

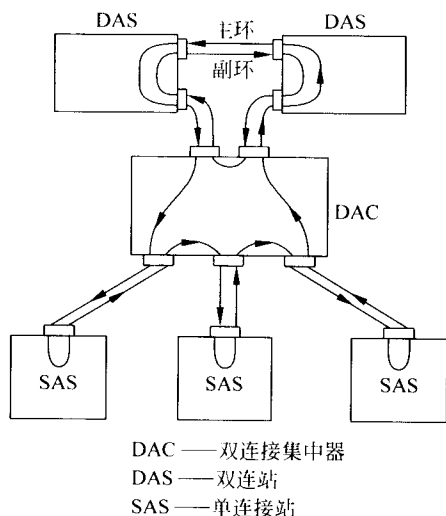


图 2 FDDI 双环路拓扑结构

表示当光纤开路和站点故障时,故障两侧的站点光旁路开关引导光信号绕过故障点直接送至副环光纤上,重新构成新的环路的情形。

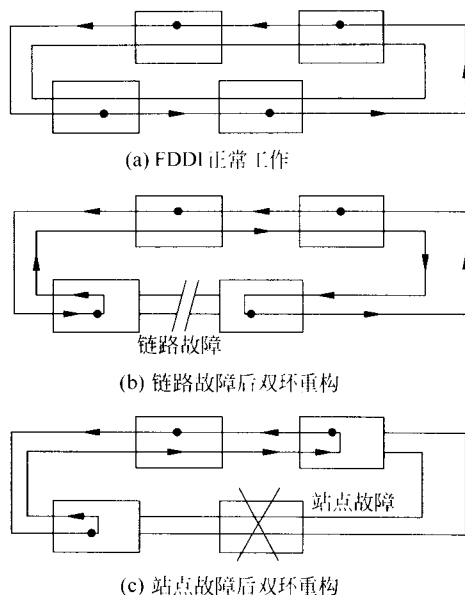


图 3 FDDI 双环重构功能

FDDI 的介质访问控制采用多权标的权标传递法,当网上所有站点都空闲时,权标在环上运行。当某一站点要发送信息时,一旦权标通过该站,就把它截下并马上向环上发送信息,发完信息后就释放权标。其他想要发送信息的站点截到该权标时,也可向环上发送信息。信息发送完毕再释放权标,如此类推。这样,就可能有多个信息在环上同时传送。信息送到目的站点被接收后再返回到发送站点,并由发送站点将其删除。

FDDI 具有速度快、传输距离长(可达 100km)和可靠性高等优点,较多用在企业主干网中,但随着异步传送模式(ATM)和快速以太网的出现,FDDI 的应用越来越少了。

参考文献

高传善等译校. 局域网与城域网(第 5 版). 北京: 电子工业出版社, 1998

(方起兴)

guangxian genzong jishu

光线跟踪技术(ray tracing technique) 通过模拟光线在环境中的走向轨迹进行图形绘制的一种全局光照技术。光线跟踪技术于 20 世纪 60 年代末

提出,用于确定阴影和可见面。70年代末,80年代初将该技术应用到**真实感图形生成**的复杂光照计算,包括反射、透射等。该技术的发展和應用显著地改善了计算机生成的真实感图形的质量,因而光线跟踪成为**计算机图形学**中具有重要地位的技术。

由环境中的物体产生真实感图形的过程是一个从三维空间向二维图形屏幕投射的过程。屏幕上每一像素所具有的色彩和亮度应精确地反映整个环境中的复杂光照在该点投影方向上所产生的综合效果。光线跟踪技术正是从这一原理出发,以观察点为起始点,通过屏幕上每一像素点沿着光线投射的逆方向去“追根溯源”,累积各种光照能量。

光线跟踪有时也可称为光线投射。但光线投射现在通常是指从视点出发,经像素向物体空间进行一次性投射。这种投射方法在该技术发展初期曾用于探测可见面。而光线跟踪一般是指可处理反射、折射等光照效果的复杂多层的跟踪技术。如图1所示,跟踪光线从视点出发通过屏幕交于物体A点,其光亮度 I 由3部分光强分量组成:光源直接照射,即由局部光照模型计算的光亮度(L_1),物体表面反射光线方向上的光亮度(R_1)以及透射方向上的光亮度(T_1)。继续沿反射方向和透射方向跟踪,又有

$$R_1 = R_{11} + T_{11} + L_2, T_1 = R_{12} + T_{12} + L_3$$

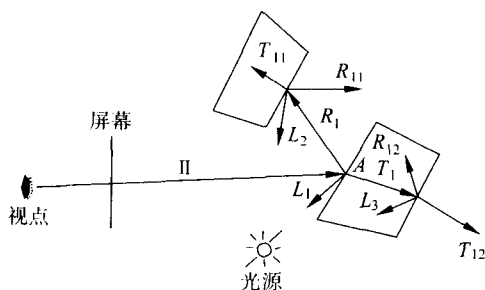


图1 光线的递归跟踪

如此继续跟踪下去,其递归跟踪过程便形成了一棵光线跟踪树。该树的层次深度根据需要而定。树的每一个结点上的光亮度为该结点的综合光亮度。实际计算时是从树的最低层即叶结点的光亮度算起,沿着光线跟踪的反方向逐层归纳各子树的光照,从而得到树根处的值,即为该像素处最终的光照效果。

由于使用光线跟踪方法很容易沿着光线求取反射及透射效果,因而该技术不仅有利于显示阴影,而且很适合于模拟高光反射即镜面反射、透视等光照

效果。

参考文献

1. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京: 科学出版社,2001
2. 孙家广,杨长贵. 计算机图形学(新版). 北京: 清华大学出版社,1995 (吴恩华)

guangxue biaoji yueduji

光学标记阅读机 (optical mark reader, OMR) 用于直接识别标志在信息卡上预定格式信息点内容的输入设备。常用的信息卡是专门设计的纸质卡片,上面有许多待涂点,称为信息点,信息点大小约为1 mm×3 mm。用笔将信息点填涂后形成黑色长方形,它与未填涂的信息点(空白)分别表示两种状态(参见图1),通过定义赋予两种不同的含义。OMR通过对信息卡上信息点的识别达到对信息卡上内容的了解。因此它具有简捷、快速、准确、高效等特点。

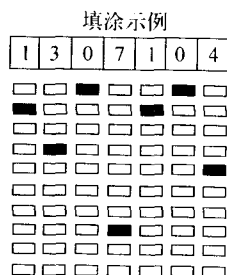


图1 信息卡(局部)

OMR的结构由机械传动、光电传感器、通信接口等三部分组成。机械传动机构驱动信息卡平稳地通过光电传感器,最后落到接纸仓内。光电传感器包括有半导体发光器件及光敏器件。发光器件将发出的光照射到信息点上,光敏器件接收其反射光。由于信息点填涂或空白两种状态的反射光强度不同,而使光敏器件产生强弱不同的电信号,经过处理达到识别的目的。通信接口用来接收计算机的控制命令并将光电传感器处理后的信息发送给计算机。

OMR主要技术指标是读卡准确度及读卡速度。读卡准确度采用误码率来衡量,误码率等于读错信息点的个数与读过信息点总个数之比。读错信息有两种情况:一种叫漏点,对已填涂过的信息点未读出;另一种叫冒点,把未涂空白点或涂后又擦掉的信息点误认为填涂点。国家教委考试中心要求招生考

试领域使用的 OMR 产品误码率应小于百万分之一。读卡速度体现了 OMR 快速输入数据的能力,因 OMR 类型及信息卡种类不同而有很大差异,大致范围是每秒读 1 至 4 张。

OMR 的类型很多,按进纸方式可分半自动型和全自动型。半自动型是靠人工将信息卡一张张送入 OMR,再由 OMR 自动读取信息卡内容。全自动型是使整个进纸与读取过程自动化,一叠上百张信息卡一次放入,OMR 可自动一张一张自动进纸,直到全部读完。按识别涂点类型可分铅笔型和通用型。铅笔型只能识别铅笔或含碳素笔的涂写点。通用型能识别钢笔、圆珠笔、铅笔等多种笔的涂点,但不能识别红色和白色。按错误处理方式可分成有分拣功能和无分拣功能两类。无分拣功能 OMR 在阅读信息卡过程中,出现错误时自动停机等待处理,而有分拣功能的 OMR 能把有错误的信息卡自动放入另外一个接纸仓内,读卡过程不停,一批读完后再次作处理。

采用 OMR 处理可以省去大量由人工键入数据的过程,能及时得到分析统计结果。OMR 在教育、卫生、交通、人事、计划生育等许多领域都得到日益广泛的应用。(徐萌)

guangxue zifu yueduji

光学字符阅读机 (optical character reader, OCR) 以光学扫描的方法,将字符以图像方式输入计算机,经过识别与处理,实现字符输入的设备。

光学字符识别技术已经过了多年的发展历程。光学扫描仪及微处理器技术的高速发展,为光学字符识别技术的实用化创造了条件。

光学字符识别的对象是字符页或其他字符媒体,经过光学扫描系统把该部分字符以图像方式输入计算机,利用算法对字符图像的切分,逐个把局部图像进行特征提取、比较、分类等字符识别算法的处理,经过与样板字库的反复比较,可以得到一个最相近的样板字元作为结果,这样就可以在屏幕上显示或传输给相应的处理单元。

典型的光学字符阅读机由三部分组成(见图 1)。

(1) 光学图像扫描仪 通过光学系统移动扫描得到高清晰的图像。

(2) 高性能微处理机以及光学字符识别软件 主要包括图像预处理器,以特征提取及分类为主的字符识别算法和提高识别率的后处理器。

(3) 输出接口及应用部件 根据用途不同,识

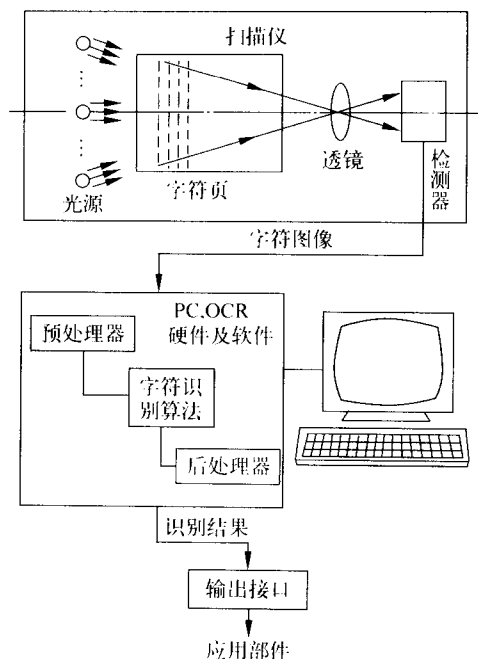


图 1 光学字符阅读机结构原理图

别结果可送往数据库、字处理器或其他自动化处理设备,例如信件自动分拣机的分类装置等。

光学字符阅读机可分成通用系统和专用系统两大类。通用系统主要用于文字识别和处理;专用系统,例如,票据识别检查机、支票兑现机、信件自动分拣机等,主要用于邮政、金融等领域。(潘弘寿)

guangzhao moxing

光照模型 (illumination model) 用于计算物体表面光照强度的数学模型。景物表面的颜色和明暗色调是表现景物的几何形状、表面材料属性以及所处环境的最重要手段。在计算机图形学中,用物理学中的光照强度表征物体辐射光能的强弱和色彩组成。为了生成真实感图形,必须根据光学原理计算物体表面各处的色彩和光照。当物体受到光源照射时,光线可以通过物体表面反射产生反射光;如果物体是透明体,光线还可穿透物体产生透射光。反射光和透射光进入人眼产生视觉效果。

一般,光照模型可分为局部光照模型和全局光照模型两大类。局部光照模型只考虑光源直接照射所引起的反射光强,模型简单,计算量较小(参见光亮亮度计算);全局或整体光照模型则全面考虑环境中物体之间光能的多重反射、透射以及相互影响。

光线跟踪技术和辐射度技术是全局光照模型两种最重要的方法。

参考文献

1. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2001
2. Foley J D, Dam A V 著. 交互式计算机图形学基础. 唐泽圣,周嘉玉等译. 北京:清华大学出版社,1986
3. Cohen M F, Wallace J R, Hanrahan P, Greenberg D P. Radiosity and Realistic Image Synthesis. New York: Academic Press Professional, 1993

(吴恩华)

guangbo luyou xuanze

广播路由选择 (broadcast routing) 向网络中所有其他站点发送报文的路径选择方法。某些应用,如天气预报、军事指挥、股市行情、现场会议等有时需要将数据发向网络中所有的站点,这就是**广播**。

广播路由的实现有许多方法。最简单的办法就是让源站点将同一报文重复发送若干遍,分别发往所有其他的站点。显然这种方法不仅浪费带宽,而且还要求源端拥有全部目标站点的完整清单。另一种方法是**洪泛法**,也称为**扩散法**,就是每个站点都将收到的报文,向除了到达的线路外的其他所有的线路转发出去,最终实现广播的目的。这种方法也会产生大量重复的报文,而且除非采取某种抑制措施,否则会产生无穷多的报文。还有一种**多目标路由选择方法**,即一个报文中可以含有一批目标站点的清单。当它到达一个目标站点后就将该站点从此清单中除去;若到达一个路由站点,则检查该清单及路由表,将从一条输出线路转发的目标集中在一个新的带多目标清单的报文中,并转发出去。最理想的方法是找到一棵以源站点为根的覆盖全网的翼展树(又称生成树),而后沿着树枝的线路采用多目标路由的方法发送。但是,实际上并不是每个路由站点都知道其在生成树上的正确位置。还有人提出了并不需要知道生成树的逆向路径转发方法。这种方法的基本思想是,当广播报文到达路由站点时,它查看该报文是否来自通常用于发送报文到广播源的线路,如果是,则此广播报文非常可能就是从源来的,应该将此报文复制转发到除进入线路外的所有其他线路;否则报文就可能是从其他地方转发过来的多余副本而被扔掉。这种方法虽然也会增加一些额外开销,但还是比较合理有效而又易于实现的。

某些应用中,一组站点协同工作构成组,报文需要在组内广播,此时我们称为**组播**。组播更为复杂,还需要有良好的组管理机制,允许站点动态地加入或退出组。

参考文献

- Tanenbaum A S. Computer Networks. Third Edition. New Jersey: Prentice Hall, 1996 (高传善)

guangpu yuyan

广谱语言 (wide spectrum language) 可在不同的抽象级别上书写软件需求定义、功能规约、设计规约及实现的语言。

好的软件规约要在较高的抽象级上说明做什么,强调明晰,而好的实现必须针对虚拟机给出如何做的详细描述,强调功效。明晰和功效很难在一个程序中同时得到满足,这两者之间有很大距离。程序(规约)在这个距离间分成若干个抽象级别。广谱语言及其相应的支撑系统能够在同一个语言架构下从明晰的规约通过推理逐步向高效的程序转化,或者提供低级程序相对于高级规约的正确性证明的辅助手段。

广谱语言应提供规约级到实现级的数据结构和控制结构的构造,并提供结构间语义等价的转换规则,用于自动转换或验证低级结构相对高级结构的正确性。其代表性的语言是 CIP-L。 (伊波)

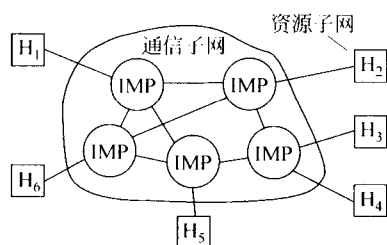
guangyuwang

广域网 (wide area network, WAN) 作用的地理范围从数十公里到数千公里,可以连接若干个城市、地区,甚至跨越国界,遍及全球的一种通信网络。又称为**远程网**。

从计算机网络的形成和发展历程来看,早期的计算机网络大多为**联机系统**。它是由多台远程终端设备通过公用电话网连接到1台中央计算机构成的,用以解决远程信息的收集、计算和处理。根据信息处理方式的不同,可以有实时处理、批量处理和分时处理等不同的联机系统。这些联机系统是计算机与通信相结合的先驱,提供了计算机通信的许多基本技术,而其本身也成为以后发展起来的计算机网络的组成部分。

20世纪70年代,美国国防高级研究计划署(DARPA)首先采用存储转发分组交换原理开发了ARPA网(ARPANET),所采用的一系列技术为计算

机网络的发展奠定了基础。此后,许多发达国家、大的企业集团纷纷建立起公用的分组交换网和专用的计算机通信网,如美国的 TELENET、加拿大的 DATAPAC、欧洲共同体的 EURNET、IBM 公司的 SNA 网、DEC 公司的 DECnet 网等。这些计算机网络通常是由通信子网和资源子网两级构成的。图 1 给出了它们的示意图。



H——主机；IMP——接口报文处理机

图 1 两级子网构成的计算机网络

虽然上述计算机网络能提供多种服务,但由于众多的网络系统没有统一的网络体系结构,要把它们互连起来,实现更大范围内的信息交换与资源共享是很困难的。计算机网络的国际化趋向成为一种必然。国际标准化组织 (ISO) 制定的开放系统互连基准 (参考) 模型 (OSI/RM) 为发展标准化网络体系结构奠定了基础。后来发展起来的广域公用通信网,如公用数据网、帧中继网和综合业务数字网等大多是遵循 OSI/RM 的,而覆盖一百几十个国家的世上最大的广域网 (WAN)——Internet 则是以 TCP/IP 协议为基础的。

参考文献

1. Piscitello D M, Chapin A L. Open Systems Networking: TCP/IP and OSI. New York: Addison-wesley Publishing Company, 1993
2. Tanenbaum A S. Computer Networks. Englewood Cliffs: Prentice Hall, 1988 (史美林)

guangyu xinxi fuwu xitong

广域信息服务系统 (wide area information server, WAIS) Internet 上的一种易于检索并可获取远程数据文档的动态超文本系统。WAIS 采用客户-服务器模式和 Z 39.50 协议标准,特别适用于基于关键字的信息查询。

WAIS 可以有 3 种使用方式: ①远程登录; ②本地运行 WAIS 客户机; ③通过 Gopher 进入 WAIS。

WAIS 检索时,会显示出一个提示表,表中将要查找的文档按相关性,从高到低顺序列出提示信息,相关性的次序是以文档中出现检索字的次数来计算的。

WAIS 允许用户搜索和检索在网上的任何数据库的信息。用户使用关键字检索 WAIS 服务器中相应文档,服务器列出含有该关键字的文档清单,用户再发送需要复制的相应文档名,即可获取所需的文档。

(胡道元)

guijie fangfa

归结方法 (resolution method) 一阶逻辑中的一个完备的推理方法。亦即对于一阶逻辑中任一恒真的公式,使用此方法都能在有限步内证明这个公式的恒真性,因此归结方法成为定理机器证明的一种重要方法。

从 17 世纪数理逻辑诞生起,人们就期望得到一个机械的逻辑方法,用这个方法能够证明用数理逻辑描述的所有定理。在长期的研究中,虽然得到了一些算法,但是这些算法在计算机上实现时,都因为过于复杂而不能真正地证明定理的任务。直到 1965 年 J. A. Robinson 提出归结方法,才使得在计算机上证明定理向前跨一大步。

下面以命题逻辑为例介绍这一方法。

我们将原子或者原子的否定统称为文字。由文字组成的析取式称为子句。例如, $A \vee B \vee \sim C$ 是子句,其中 A, B, C 是原子。 $(A \wedge B) \vee C$ 不是子句。

归结式: 设 C_1 和 C_2 是两个子句。若在 C_1 和 C_2 中分别存在两个文字 L_1 和 L_2 , 并且 $L_1 = \sim L_2$, 不妨设

$$C_1 = L_1 \vee C'_1$$

$$C_2 = L_2 \vee C'_2$$

其中 C'_1 和 C'_2 也都是子句,则 $C'_1 \vee C'_2$ 称为对 C_1 和 C_2 使用归结方法得到的归结式。

可见,归结方法是一种操作起来非常简洁的方法: 在两个子句中删掉互补的两个文字 (如果存在的话), 将剩下的子句再析取起来。例如, $C_1 = \sim A \vee B$, $C_2 = A$, 则对 C_1 和 C_2 使用归结方法, 得到 B 。而 C_1 也可等价地写成 $(A \rightarrow B)$, C_2 是 A , 由 C_1 和 C_2 得到 B 的推理实际上是三段论推理。因此,三段论是归结方法的一个特例。

用归结方法证明数学定理实质上是一个反驳过程。在数理逻辑中,一个数学定理,通常写成如下公式:

$$A_1 \wedge A_2 \wedge \cdots \wedge A_n \rightarrow B$$

其中 A_1, \dots, A_n 是前提公式, B 是结论公式。证明上述定理成立, 相当于证明公式 $(\sim A_1 \vee \dots \vee \sim A_n \vee B)$ 是恒真的, 也相当于证明如下公式:

$$\begin{aligned} & \sim (\sim A_1 \vee \dots \vee \sim A_n \vee B) \\ & = A_1 \wedge \dots \wedge A_n \wedge \sim B \end{aligned} \quad (*)$$

是恒假的。将公式 $(*)$ 等价地化成合取范式, 得到一个子句集 S (S 中子句之间是合取关系), 用归结方法证明了 S 的恒假性, 就相当于证明了上述定理。例如, 要证明如下定理:

$$((A \rightarrow B) \wedge (B \rightarrow C) \wedge A) \rightarrow C$$

只要证明公式 $((A \rightarrow B) \wedge (B \rightarrow C) \wedge A \wedge \sim C)$ 是恒假的即可。将此公式化成合取范式:

$$(\sim A \vee B) \wedge (\sim B \vee C) \wedge A \wedge \sim C$$

将上面公式写成如下子句集 S :

$$S \begin{cases} (1) & \sim A \vee B \\ (2) & \sim B \vee C \\ (3) & A \\ (4) & \sim C \end{cases}$$

(1) 和 (3) 作归结得到 B , B 和 (2) 作归结得到 C , C 和 (4) 是一对矛盾, 作归结得一恒假子句 (称为空子句), 因此 S 是恒假的, 从而证明了定理。

如此简洁的方法却有着很强的功能。

完备性定理: 子句集 S 是恒假的, 当且仅当从 S 出发, 使用归结方法可推出一对矛盾 (即得到一个空子句)。

这一方法不仅对命题逻辑适用, 对一阶逻辑也适用, 当然需要加进合一替换的概念。

归结方法也正在被引进各种非标准逻辑中, 成为自动推理领域中的一种基本方法。

参考文献

1. 刘叙华, 姜云飞. 定理机器证明. 北京: 科学出版社, 1987
2. Chang C L, Lee R C T. Symbolic Logic and Mechanical Theorem Proving. New York: Academic Press, 1973 (刘叙华)

guina tuili

归纳推理 (inductive inference) 由特殊性事例推导出一般性原理的推理方法, 或者, 由一般性程度较低的知识提高到普遍性程度较高的知识的过程, 是人类典型的逻辑思维形式之一。

归纳推理的前提是一些个别事物的单称判断, 结论是关于一类事物的全称判断。因而归纳推理是

不保真的, 它的结论往往带有某种程度的或然性。也正因为如此, 归纳推理的结论常会超出前提所含的知识, 从而使得它成为知识获取、机器发现及许多机器学习方法的基础。

归纳推理的形式有多种, 常用的有枚举归纳推理、数学归纳法、科学归纳推理、探求因果联系的归纳推理和反绎推理等。

枚举归纳推理 包括简单枚举归纳推理和完全归纳推理。简单枚举归纳推理的模式如下:

$$\begin{aligned} & s_1, s_2, \dots, s_n \text{ 都具有属性 } p \\ & \underline{s_1, s_2, \dots, s_n \text{ 是 } S \text{ 类中的部分对象}} \\ & S \text{ 类中对象都具有属性 } p \end{aligned}$$

如果在简单枚举归纳推理的前提中将考察的对象全部列举出来, 那么它就成为完全归纳推理。由于完全归纳推理结论中的知识是完全包括在前提知识中的, 所以只要前提完全真实, 它的结论就必然是真实的。从这一点讲它又具有演绎推理的特征。

枚举归纳推理是基于事例的学习和一些机器发现系统的逻辑基础。在大多数学习系统中它们都体现为“推广规则”。

数学归纳法是数学中常用的一种定性证明方法, 它要求归纳的对象按某种序关系构成严格的序列。数学归纳法实际上也是一种完全归纳法, 其推理模式如下:

$$\begin{aligned} & \text{序列 } S \text{ 的第一个成员 } a_1 \text{ 具有属性 } p \\ & \text{如果 } S \text{ 的成员 } a_k \text{ 具有属性 } p, \text{ 则 } S \text{ 的成员 } a_{k+1} \text{ 也具有属性 } p \\ & \underline{S \text{ 的所有成员具有属性 } p} \end{aligned}$$

科学归纳推理 是根据对部分对象及其属性间的因果关系或其他必然联系的认识, 从而对该类所有对象做出一般性结论的推理, 其推理模式为:

$$\begin{aligned} & s_1, s_2, \dots, s_n \text{ 都具有属性 } p \\ & \underline{s_1, s_2, \dots, s_n \text{ 是 } S \text{ 类的部分对象}} \\ & \text{且 } S \text{ 与 } p \text{ 有必然联系} \\ & \underline{S \text{ 类的对象都具有属性 } p} \end{aligned}$$

虽然科学归纳推理和简单枚举归纳推理都是不完全归纳推理, 但是两者之间有明显的差别。科学归纳推理是建立在对前提中的事物与属性间因果关系的科学分析基础之上的, 因而其结论的可靠程度取决于前提中事物与属性之间因果相关程度的强弱, 而与事例的数量无关。

探求因果联系的归纳推理是一类推理方法的统称。研究事物之间的因果联系是进行科学归纳推理的必要条件,而认识事物之间的因果联系是一个复杂的过程。但是有些较为简单的又具有一定普遍意义的确定因果联系的方法还是可以总结出来的,统称为探求因果联系的归纳推理方法。最著名的有穆纳五法,即求同法、求异法、求同求异并用法、共变法和剩余法。它们的模式分别如下:

求同法	求异法
$A, B, C—a$	$A, B, C—a$
$A, D, E—a$	$\neg, B, C—$
...	
A 是 a 的原因	A 是 a 的原因
求同求异并用法	
有 A $\begin{cases} A, B, C—a \\ A, D, E—a \\ \dots \end{cases}$	无 A $\begin{cases} B, M, N— \\ D, O, P— \\ \dots \end{cases}$
	A 是 a 的原因
共变法	剩余法
$A_1, B, C—a_1$	已知现象 abc 的复合原因是 ABC
$A_2, B, C—a_2$	又知 B 是 b 的原因
$A_3, B, C—a_3$	C 是 c 的原因
A 是 a 的原因	A 是 a 的原因

其中 A_1, A_2, A_3 是 A 的不同变化;
 a_1, a_2, a_3 是 a 的不同表现。

反绎推理是以某种演绎推理为存在条件的归纳推理。比如有三段论演绎推理:

$$\frac{\text{如果 } p, \text{ 则 } q}{p} \\ q$$

对应地有反绎推理:

$$\frac{\text{如果 } p, \text{ 则 } q}{q} \\ p$$

反绎推理已成为许多决策系统、故障诊断系统的基本推理方法。

随着人工智能应用对知识获取技术越来越迫切的要求,以及以演绎推理为基础的自动推理技术的逐步成熟,人们开始更多地将注意力转向包括归纳推理在内的各种机器学习技术和方法的研究。

参考文献

1. 中国人民大学哲学系逻辑教研室. 形式逻辑. 北京: 中国人民大学出版社, 1984

辑. 北京: 中国人民大学出版社, 1984

2. Brozdlil P B. Machine Learning: ECML—93. Berlin: Spring-Verlag, 1993 (赵沁平)

guina xuexi

归纳学习 (inductive learning) 以归纳推理为基础的一种机器学习方法。它是从事物的部分知识学习事物的整体知识,从个别事物的知识学习一类事物的知识的过程。

归纳学习起源于 19 世纪的哲学和心理学研究, 20 世纪中叶, 人工智能领域开始对其进行研究。E. B. Hunt 是用计算机程序模拟人类归纳学习的前驱之一, 他在 1962 年至 1966 年间研制了从 CLS-1 到 CLS-9 的一系列概念学习系统。70 年代中期归纳学习的研究逐步发展, 研制出一批归纳学习系统, 如 ID3 和 AQ11 等。进入 80 年代以后, 归纳学习成为机器学习研究的焦点之一。

归纳学习中得到的知识称为归纳断言。由于归纳学习中使用的归纳推理规则是不保真的, 所以归纳断言的证实要由人或其他演绎系统来完成。归纳学习过程可以描述为

已知: ① 数据集 Δ : 一组来自实验或观察的数据、例子;

② 假说空间 H : 以某种语言描述的归纳断言的集合;

③ 背景知识 Γ : 对数据和归纳断言的约束。

求: 归纳断言 φ , 要求:

① φ 与已知信息一致, 即 $\Gamma \cup \Delta \models \sim \varphi$;

② φ 能解释数据, 即 $\Gamma \cup \{\varphi\} \models \Delta$ 。

数据是归纳的素材, 应该是无二义性的, 搜索数据集的目的在于选择合适的数据, 以便证实或否定假说。数据集的搜索方法往往与假说空间有关, 常用的方法有: ① 选择最有利于划分假说的例子, 以便尽快缩小寻找归纳断言的范围。② 选择最有希望的假说, 再选择适当的数据来证实它。③ 选择否定某些假说的数据。

与假说空间有关的问题是对假说空间的要求和假说空间的搜索方法。对假说空间的要求有两方面: 假说空间包括归纳断言; 假说空间的表示方法不仅应该适合归纳推理, 而且要与例子的表示一致。前者关系到能否学习到要求的归纳断言, 后者影响归纳学习过程的难易程度。假说空间的搜索有数据驱动和模型驱动两类方法。数据驱动方法的优点是可以边接受数据边学习, 缺点是对数据噪声敏感。

模型驱动方法的优点是抗干扰性好,缺点是难以进行逐步学习。

由于对任意给定的数据集都能产生无限多的归纳断言,因而有必要根据需要提供一些准则,以便选择较为理想的归纳断言。这些准则是背景知识的重要组成部分。此外,这些准则对于限制归纳过程中搜索的假说空间的大小和结构起着重要作用,是提高归纳学习效率的关键。

一般说来,归纳学习分为实例学习、观察与发现学习。实例学习也称为概念形成,其输入是一组正反例,任务是确定概念的一般描述。当前实例学习的研究集中在两个方面:例子—类型一般化,部分—整体一般化。前者的输入是某类对象的独立实例,目标是归纳出这类的一般描述。后者的输入是对象的局部,任务是假设整个对象的描述。

在实例学习中施教者提供正反例并进行分类,因此它是有示教的归纳学习。而观察与发现学习是从未经分类的观察中学习,因此是无示教的归纳学习。

在实例学习中,已知概念的正例和反例描述,学习系统归纳出概念的描述。实例学习是人工智能研究者最早研究的归纳学习类型,现已发展成为机器学习领域中成果最丰富的分支,并且它在某些领域的应用已成为机器学习走向实用的先导。按学习任务的复杂程度实例学习可分为学习单个概念和学习多个概念。

虽然学习单个概念是最简单的学习任务,但它是其他学习策略的基础。学习单个概念问题可以描述为

已知:① 概念描述语言。它将假说空间隐含定义为该语言可以表示的所有规则;

② 某概念的正例和反例集合。

求:假说空间的一条规则,该规则能覆盖所有正例而排除任一反例。

J. R. Quinlan 的 ID3 算法是决策树方法的典型代表,它继承并改进了 E. B. Hunt 的概念学习算法 CLS 的思想,能学习单个概念。

实际应用往往要求学习系统能归纳出多个相互独立的概念。R. S. Michalski 等人的归纳学习系统 AQ11 是一般化—具体化策略的典型代表,它用于多个概念的学习。AQ11 把学习分类描述问题转化为一组学习单个概念问题。为了寻找某一类事物的描述,它把该类的例子作为正例,而把其他类的例子作为反例,由此找到覆盖全部正例而不覆盖任一

反例的描述作为某一类的分类规则。

在观察与发现学习中,环境提供的观察是未经分类的例子,施教者只提供少量初始知识,系统需要产生解释观察的假说。按照与外部环境交互作用的程度又可将其分为两类。一类是被动观察,学习者被动接受各种观察并给出分类。另一类是主动试验,学习者扰动环境并观察结果。

观察与发现学习是以系统的初始知识为基础,从环境提供的观察数据中学习。一般说来这种学习比较复杂,除使用归纳外,还需要组合演绎、类比和反绎推理。这方面的研究内容很广,涉及概念聚类、构造性归纳学习,定性定量发现,概念形成和修改等。

概念聚类就是把观察的事物形成分类并建立相应分类体系的过程,可描述为

已知:① 对象及其特征描述集合;

② 背景知识(包括问题约束,特征的性质及评价聚类质量的准则)。

求:对象的分类体系。每类由一个表达式描述,任一类型的各子类的描述是逻辑上不相交的,分类体系应使聚类质量准则最优。

构造性归纳学习属于知识密集型学习,开始时系统中有预先确定的概念、知识结构以及与领域有关的启发式规则、约束和变换。在学习过程中系统通过对已有知识的交换和修改,产生新的属性和概念。因此,这类学习系统一般是为专门领域开发的,不能直接用于其他领域。通常把从数据中抽取逻辑结构的算法称为 0 级算法,把能创建可用于重组织知识的发现有用新概念的算法称为 1 级算法。这样,构造性归纳学习算法是 1 级算法,其余归纳学习算法均是 0 级算法。D. B. Lenat 的数学概念学习系统 AM 是有代表性的构造性归纳学习系统。

定量发现(发现数值间的关系)在科学发现中占有重要地位,有关研究工作始于 P. Langley 等的 BACON 系统——物理定律重新发现系统。BACON 的运行过程是反复检查数据,并用改进操作来产生新的项,直到发现一个项总是常量,这时就把学习到的概念表示为“项 = 常量”的形式。

归纳学习已经引起人们越来越多的关注,并取得了令人瞩目的进展,其应用范围正在不断扩大。近年来又出现了归纳式程序设计等多种新的研究途径,呈现出渐进、持续的发展势头。总之,归纳学习是一个充满活力、具有广阔应用前景的研究方向。

参考文献

1. 石纯一, 黄昌宁, 王家祯. 人工智能原理. 北京: 清华大学出版社, 1993
2. Forsyth R. Machine Learning: Principles and Applications. London: Chapman and Hall Ltd, 1989
3. Liu W Z, White A P. A Review of Inductive Learning. In I. M. Graham and R. W. Milne (eds.) Research and Development in Expert System VIII. Cambridge: Cambridge University Press, 1991

(赵沁平 李波)

guina zonghe fangfa

归纳综合方法 (inductive synthesis method)

借助反映程序性态的实例, 利用**归纳推理**导出程序的方法。简称归纳方法。该方法的特点是用户通过反映目标程序输入/输出行为的实例来指明需求。由于用户所给实例集只能反映目标程序的部分性态, 所以用实例作为问题规约是不完备的。归纳程序综合系统必须能对被指明的性态作出合理假设, 而不是仅仅产生能覆盖实例的程序。

基本方法 主要有**实例法**和**轨迹法**。

实例法 借助给定的一组输入输出实例, 逐步推导出适于一类问题的程序。实例法大都以 LISP 作为目标语言, 其中 P. D. Summers 归纳程序系统的工作最具代表性。他给出了一个基于简单递归模式的 LISP 程序推理的一般方法和匹配过程, 用于在多个输入输出实例中找出递归关系并加以推广。该方法的缺点是, 由于仅使用结构匹配技术, 过分依赖于形式而未考虑语义信息, 当输入、输出对之间差异较大时, 系统要直接认识出其内在联系相当困难。实例的执行轨迹是沟通输入与输出之间联系的一种手段。

轨迹法 通过所给实例的执行轨迹, 逐步推导出程序。由一组轨迹综合出程序的方法是: 首先通过试验方法生成可能的程序。若所生成的程序能解释轨迹, 那么它即为所求。然而, 由用户提供程序执行轨迹却十分困难, 如果某领域已完全公理化, 便可利用问题求解程序自动产生轨迹。

上述方法都是根据一定的程序模式, 通过匹配发现递归关系而推导出程序。

方法改进 为便于实用, 出现了基于归纳推理的程序假设空间构造的归纳方法。

结构问题 将假设空间组织成一复杂结构, 而非简单的线性表结构。其目的是, 当发现一不相容

实例时, 能够删去多个假设; 且有助于搜索与实例相容的假设。

一种简明结构是用一阶谓词中的蕴涵关系所构造的假设空间。蕴涵关系依据从一般到具体的次序给出了假设空间上的一种偏序, 即如果 A 蕴涵 B , 则称 A 比 B 更一般, 或 B 比 A 更具体。在此结构下, 若假设 A 与某个正例不相容, 则无需再考虑比 A 更具体的假设。反之, 当 A 包括了一反例, 则无需再考虑比 A 更一般的假设。利用该结构的一成功算法是用于概念学习的“版本空间”算法。其主要思想是, 分别从最一般和最具体的两端逐步向正确假设逼近。它使用两个假设集合, 一个是由最一般假设构成的集合 G , 它不含任何反例; 另一是由最具体假设构成的集合 S , 它包括所有正例。若版本空间因实例(正例和反例)的增加而变小, 且最终收敛于一个假设, 则即为学习结果。

条件问题 在搜索假设空间时, 人们期望虽无法完全正确地给出所需假设应具备哪些性质, 但能明确指出它不该满足哪些条件, 即假设的必要条件问题。一种方法是采用诊断技术: 当发现假设与实例不相容时, 利用诊断技术找出不相容的原因, 此即为某个非法条件。一旦发现, 则在以后的搜索中不再考虑具有这种非法条件的假设。

为了防止假设空间的组合爆炸, 并缩小搜索范围, 可以从所要求假设应满足的充分条件入手, 给出一组判定标准。这种方法有时能显著缩小搜索范围, 甚至只要找出一个假设即可。由于它是构造性的, 其效率依赖于刻画充分条件的简明程度。

从方法学上看, 尽管归纳方法非常诱人, 但实现技术颇难, 目前尚难以推广到具有现实复杂性的应用领域。从发展上看, 应在归纳程序综合的背景知识、程序的层次构造以及归纳假设的证实等方面进一步开展研究。

参考文献

1. 徐家福, 陈道蓄, 吕建, 王志坚. 软件自动化. 北京: 清华大学出版社, 1994
2. Smith D R. A Survey of the Synthesis of LISP Programs from Examples. International Workshop on Program Construction. Bonas, 1980

(王志坚 张家重)

guiyueji

归约机 (reduction machine) 一种面向函数式程序设计语言的非传统计算机。

冯·诺依曼计算机所用的命令式程序设计语言(例如 FORTRAN, PASCAL 等)难以编写、验证和重用程序,不适合并行处理,因而有人提出函数式程序设计语言(参见函数式程序设计语言)。这种语言有利于保证程序各部分的并行执行,更适合于编写高度并行系统结构的并行处理程序。

归约机的处理对象是表达式,其处理过程是表达式的归约过程。归约的计算模型是把函数式程序看作嵌套的表达式,它的执行过程就是求出表达式的值来代替表达式本身。表达式的求值,是由一系列表达式归约组成,而表达式一次归约是一次函数应用,或是一个子表达式的变换。例如,内积的定义为:

$IP = (+ (AA, *), TR)$, 求数组 $(1, 2, 3, 4)$ 和 $(5, 6, 7, 8)$ 的内积的归约过程如下:

- (1) $\langle IP, ((1, 2, 3, 4), (5, 6, 7, 8)) \rangle$
- (2) $\Rightarrow \langle (+, (AA, *), TR), ((1, 2, 3, 4), (5, 6, 7, 8)) \rangle$
- (3) $\Rightarrow \langle +, \langle (AA, *), \langle TR, ((1, 2, 3, 4), (5, 6, 7, 8)) \rangle \rangle \rangle$
- (4) $\Rightarrow \langle +, \langle (AA, *), ((1, 5), (2, 6), (3, 7), (4, 8)) \rangle \rangle$
- (5) $\Rightarrow \langle +, (\langle *, (1, 5) \rangle, \langle *, (2, 6) \rangle, \langle *, (3, 7) \rangle, \langle *, (4, 8) \rangle) \rangle$
- (6) $\Rightarrow \langle +, (5, 12, 21, 32) \rangle$
- (7) $\Rightarrow 70$

一个操作符及其变元的组合称为可归约表达式。在嵌套的多个表达式中已计算出其变元值的可归约表达式称为最内层可归约表达式。归约过程就是将每个最内层可归约表达式用其值来替换,这样又形成了新的最内层可归约表达式,然后再对其进行归约。最后,整个程序全部被归约,仅留下最终结果。

从系统结构看,归约机是一种以需求驱动方式进行操作计算机。在它的控制图中,一个节点上的操作仅在需要用到它的输出结果时才开始启动,如果此时这个节点的输入数据还未能获得,就由这个节点再去启动能获得它的各个输入数据的节点,就这样把需求链一直延伸下去,直到遇到常数为止。然后再按反方向逐级进行操作,把输出结果返回给需要它的上一级节点,直到最初发出需求的节点为止。需求驱动只计算那些在获得最终结果时需要其输出的节点,它执行的是最低限度的计算工作,它比数据流计算机工作效率高,且节省资源。

归约可分为串归约和图归约两类。在串归约机中,表达式、函数定义和目标以字符串形式存储,函数式程序可以不经翻译直接输入串归约机中处理。在图归约机中,表达式、函数定义和目标以图的形式存储,处理的对象是图。串归约总是优先处理最内层可归约表达式,当多个操作要求使用同一个数值时,计算该值的公共子表达式要为每个操作分别求值,形成该值的多份副本送往不同的操作,这种“按值调用”机制会使系统的使用效率降低。图归约实行“按引用调用”机制,一旦计算出一个表达式的值,就保存起来供再次使用,其他操作要求使用该值时,不必重新求值,这样可以减少重复计算工作量,但要能访问已计算好的共享数据,就要添加指针和存储地址,增加复杂性。

归约机是在 20 世纪 70 年代末开始研究的,但仅对其系统结构做过一些试验性的工作。在有影响的项目中,串归约机有美国北卡罗来纳州立大学的细胞树机等,图归约机有 ALICE 系统、GRIP 系统和 Flagship 系统等。

(孙强南 郑纬民)

guifanhua

规范化(normalization) 把非规范模式(关系模式)分解为两个或多个规范模式的过程和方法。

规范化的方法是把大关系模式分解成小关系模式,把大模式中对非键码属性集的依赖变成小模式中对键码的依赖,从而消除关系模式内的不合适的数据依赖,达到简化数据库的修改和减少数据冗余的目的。例如关系模式 $R(\text{工号}, \text{工资级别}, \text{工资额})$ 中工号是键码,工资额函数依赖于工资级别而后者不是键码,所以不是第三范式的关系模式。按此模式构造的数据库中,工资级别与工资额的对应关系有大量重复(有许多职工工资级别相同因而工资额也相同)且不完整(有的级别没有职工因而也没有工资额),若改成 $R_1(\text{工号}, \text{工资级别}), R_2(\text{工资级别}, \text{工资额})$, 则两者都是第三范式,用此模式构造的数据库中消除了数据冗余,也完整地反映了工资级别和工资额的对应关系。一般情况下这种分解不止一种,重要的是分解过程中不能丢失信息,即根据分解后的关系能还原成分解前的关系。

参考文献

Ullman J D. Principles of Database Systems (Second Edition). London: Pitman Publishing Ltd., 1987

(楼荣生 朱扬勇)

guocheng biaooshi

过程表示 (procedure representation) 一种与推理相结合的知识表示方法。知识表示就是求解程序。知识的过程表示有两种含义,第一种含义是,把解决一个问题的过程描述出来,称之为解题知识的过程性表示;第二种含义是,把客观事物的发展过程用某种方式表示出来,通常应用于理解以自然语言书写的故事,因此称之为故事知识的过程性表示。在某些情况下这两种含义很难截然分开。例如,在解决计划制订的问题上,解题系统根据用户提出的问题,把求解此问题的过程(解题计划)提供给用户,在此意义上可以算是前一种含义;但因为它描述的是一系列的事件,所以也可从后一种含义上去理解。

过程表示方法表示的知识可以分为两大类:一类是直接解题有关的知识,它们往往表示为一套解决某些标准问题的过程。普通程序设计语言中的一个标准函数、标准过程及产生式系统中的一个产生式,都是这种知识;另一类是指挥上述知识去解题的知识,即控制知识。表达控制知识的过程,按其表达形式的级别高低,可以分成策略级控制、语句级控制和实现级控制三大类。

采用过程性知识表示的系统,解题的具体过程和算法是直接面向用户的,用户可选用算法、修改算法,甚至可以自己设计算法。

下面给出一个用过程表示方法表示知识的例子。假定知识库中有如下知识:①人是一定会死的;②狗也一定会死的;③小张是人;④小李是人。

用过程表示的知识库由如下过程组成:

Procedure person (x)

```
if ( $x$  = 小张) or ( $x$  = 小李) then return true
else return false
```

Procedure mortal (x)

```
if person ( $x$ ) then return true
else if dog ( $x$ ) then return true
else return false
```

在这个知识库中,如果想要询问小李是否一定会死,则只需执行过程 mortal(小李)就可得出结论。

通过例子可以看出,采用过程表示的知识库具有如下特征:它是一组过程集合,对它的修改是通过增加或删除过程或修改过程来完成的。

过程表示的优点:①有利于表示启发式知识;②能实现扩充逻辑推理(如非单调推理、不精确推理等);③不需要推理机,效率高。

过程表示的缺点:①由于知识隐含在过程中,因此难于修改和证明;②固定的控制信息限定了其他可能的方法。一个过程中,由于问题求解活动隐含的方向性,使得一些没有次序的解题方法添加了不少不必要的隐含的次序控制信息;③采用过程表示的系统使用了启发式推理知识,这些知识本身是不完备的,有时在条件充分的情况下也不一定得出最优的答案,有时还不能保证推理过程是正确的。

参考文献

1. 陆汝钤. 人工智能(上册). 北京: 科学出版社, 1989

2. 管纪文, 刘大有等. 知识工程原理. 长春: 吉林大学出版社, 1988 (刘大有 唐海鹰)

guocheng kongzhi jisuanji

过程控制计算机 (process-control computer)

接受来自受控过程的信息,按一定的控制算法进行处理,及时地作用于过程,使过程得以正常进行的计算机。这里,受控过程主要指的是诸如炼油、化工、造纸、水泥、电力、冶金、纺织等连续性生产过程以及船舶、飞机的航行过程等。

对于大型工业生产,过程控制计算机是实现安全、高效、优质、低耗的大规模生产的重要保证。过程控制计算机的任务已从测量和控制工艺流程的参数发展到产品质量在线监测与控制,设备运行状态的监测、诊断和事故处理,设备之间的协调控制和连锁保护,生产状态的监控,厂级生产管理决策等。

发展历程

20 世纪 60 年代以后,开始将计算机用于过程控制。大体上经历了 3 个阶段:①60 年代中期以前是试验阶段。50 年代初,在化工生产中首先实现了数据采集和数据处理。由于当时的计算机速度较慢、可靠性差、体积大,没有直接参与过程控制,仅起了对整个生产过程的集中监视,指导生产过程控制及确保生产过程安全的作用。随着计算机运算速度加快和可靠性提高,60 年代初出现了直接数字控制 DDC 系统。在闭环控制系统中,计算机对过程参数进行巡回检测,并依据预先设定的控制参数进行处理,然后发出控制命令直接控制执行机构。②从 60 年代中期到 70 年代初期,随着小型计算机性能价格比的不断提高,在过程控制中的应用得到了很大的发展。但这个阶段仍以计算机集中控制系统为主。在高度集中控制时,为提高控制的可靠性,虽可采用

双机工作方式,但投资相应增加,故难以得到推广应用。③随着70年代开始的微型计算机的发展,计算机控制也突破了集中控制格局而发展为分级控制。所谓分级控制,是指在受控过程的各个环节直到各项具体装置或设备,都分别用微型计算机实现小范围的局部直接控制,包括一定条件下的局部最优控制。同时采用较为高档的微型计算机系统,通过计算机网络,对所有这些分散的局部控制计算机进行监督控制。如此构成一个二级控制系统。其中起监控作用的计算机称为监控计算机,俗称“上位机”。它依据过程信息和其他参数对过程进行分析和计算,自动地改变给定值,使过程满足某一性能指标(如能耗低、效率高、时间短等)。当局部直接控制计算机出现故障时,还可由上位机完成相应的控制功能,这样可以提高系统整体的可靠性。从70年代中期到80年代,进一步发展为全分布式控制系统或分布式微处理机控制系统。它综合了计算机、控制、通信和荧光屏显示技术。1987年提出了体现开放概念的智能自动化系列I/A Series和MOD 300多回路结构系统。这些系统参考了国际标准化组织推荐的开放系统互连模型,符合制造自动化协议(MAP),方便用户使用和软件移植,为计算机过程控制系统的规范化奠定了基础。

技术特点

过程控制计算机本身的工作原理和常规的计算机没有根本的区别。但针对过程控制的特殊要求,过程控制计算机也有其区别于常规通用计算机的特点,主要有以下几方面:

(1) 总线结构 过程控制计算机面向不同行业的受控过程,为满足多方面用户的控制要求,所用的计算机模块和设备应具有良好的兼容性,经不同的组合可配置成各种用途的计算机系统。为此,需要采用标准总线。计算机系统内部总线有STD总线、PC总线等。计算机系统外部总线有RS-232C, RS-422-A, RS-485, IEEE-488总线等。STD总线是一种工业标准总线,全部56根引线都有确切的定义。STD总线的特点是模板小型化(模板尺寸为165 mm × 114 mm)、系统开放式、总线兼容式,可支持8位、16位微处理器。RS-232-C总线定义了机械特性标准和电气特性标准。为提高抗共模干扰能力,RS-422-A总线采用光电隔离20 mA电流环接口电路,RS-232-C接口也采用了光电隔离。计算机各部件采用模板化结构,通过总线把各模板连接起来。常用的模板有中央处理器卡、多功能模拟量

I/O卡、电机控制卡、计数卡、多功能数字量I/O卡、光电输入继电器输出卡、信号放大及多路转换卡、接线端子板及安装架、ADAM-3000/4000/5000和工业用通信卡RS-232, 422, 485和IEEE-488等。

(2) 过程输入输出通道 为了实现对过程的控制,按要求的方式,将过程的各种物理参数送入计算机,计算机经过处理后,将控制命令作用于过程。所以,在计算机和受控对象之间必须设置过程输入输出通道,如模拟量输入输出通道、开关量输入输出通道等。对于实时采集、实时处理和实时控制的系统,还要求在保证精度的条件下,具有有效的实时多路处理功能。

(3) 适合于工业生产环境通信网络技术标准 随着现代化工业生产规模不断扩大和控制管理的要求日益提高,在工业生产环境中使用了计算机通信网络。通过它把不同地理位置、不同功能的多台计算机或设备连接起来,达到高可靠性和快速实时响应的目的。为适应工业生产环境,制定了生产过程控制计算机局域网协议,主要有制造自动化协议MAP和过程数据高速公路PROWAY两种。国际电工委员会(IEC)于1996年提出了集散控制系统网络的标准体系结构。该体系结构分为3级:第一级为现场总线,第二级为PROWAY,第三级为MAP局域网。它们的功能分别是多点连接众多的可编址I/O装置,连接过程控制器和局部操作站,连接中央操作站、控制管理计算机和生产管理计算机,构成一个计算机集散过程控制系统。

(4) 人机接口技术 操作人员和计算机是通过人机接口互通信息的,尽管是在计算机控制下运行,操作人员仍需根据受控过程的状况及时地向计算机发出各种指令,而计算机也应当能实时地显示受控过程状况和操作结果等信息。为此,需要有各种特定用途的外围设备,例如回路操作和显示、键盘和可编程键盘、发光二极管显示器、指示报警装置和荧光屏显示器等。

(5) 可靠性技术 在过程控制计算机工作的现场,通常总是存在多种形式的干扰源。因此,过程控制计算机必须强化抗干扰措施,以保证在现场环境中能正常运行。此外,为提高系统的抗干扰能力,还应重视供电技术和接地技术。因为受控过程现场环境往往十分恶劣,例如高温、潮湿、粉尘、振动、化学腐蚀等,所以过程控制计算机必须加强抗震措施、采用抗恶劣环境的机箱等。为确保受控过程的安全和可靠运行,故障诊断和容错技术在控制中的应用越

来越广泛。信号报警系统是故障诊断的初级形式,更高的要求是诊断故障的性质、内容和地理位置。容错技术是当系统中某一环节出现故障时,系统仍能维持一定的功能,使系统性能指标保持在一定的范围内。

(6) 控制算法及其参数选择和工程实现 按偏差的比例(P)、积分(I)和微分(D)进行控制的PID控制是应用最广泛的常规控制算法。用计算机实现PID控制算法,不仅仅是把PID计算数字化,而且与计算机的逻辑判断功能结合起来,可对标准PID算法进行修正以适应实际的需要。在计算机中,数字式PID控制算法往往编制成公用子程序形式。为共享该子程序,需要给每个PID控制回路配一内存数据区以便存放各个控制回路和输入输出通道的数据。数字PID控制参数的选择,除选择比例增益 K_p 、积分时间 T_i 、微分时间 T_d 和微分增益 K_d 外,还要选择采样周期 T 。除PID算法外,尚有参数最优化的低阶控制算法、惯性因子法、达林控制算法和预测控制算法等。现代控制理论的某些高级控制算法在上位机协调控制中已逐步获得应用。同时,智能控制技术,如自适应、自学习、模糊控制、神经网络控制、实时专家控制等,也已获得日益广泛的应用。

(7) 系统软件技术 除了程序开发、运行、维护所必需的常规软件环境、工具和相应的支撑软件外,由于在线控制计算机是伴随着受控过程不间断地进行检测、监控,所以必须配备实时操作系统,还应有各种针对性的中断处理程序。此外,针对过程控制的特殊性,除了常规的汇编语言、高级语言外,有时需要采用特定的语言来编制程序。因此,需要有相应的编译程序或解释程序等。

展 望

过程控制计算机将沿着上述的微型化、模块化和组合化、开放化、网络化、智能化等方向继续发展。由于过程控制计算机涉及多门学科、综合了多种技术,因此也必将随着相关学科技术的发展而发展。

(李培德)

guocheng shixian fangfa

过程实现方法 (procedural implementation method) 借助过程化手段将软件规约转化成目标程序的方法。简称过程方法。如果对应软件规约中的各个成分,其转换目标的相应成分明确,而且相应的转换映射也明确,该映射可借助过程来实现。

过程方法的实现基本类似古典高级语言的编译技术,它是软件自动化中较为成熟的一个实现途径。

软件自动化的过程方法是计算机软件工作者早就追求的目标,第一个自动化系统就是1956年的FORTRAN编译程序。至今,人们仍试图通过甚高级语言的编译程序来实现这一目标。对甚高级语言而言,由于它含有抽象级别较高的语言成分,传统的编译程序不能有效地将它们直接编译成目标代码。为此,必须放宽对传统编译的限制,即除源程序外,程序人员还可以提供附加的建议或说明,例如,对抽象级别较高的语言成分选择合适的数据表示,这种方式称为**扩展编译**。过程方法目前所采用的技术实质上就是扩展编译技术。扩展编译与传统编译接近之处在于:编译本身能自动进行,且产生的目标代码是可执行的。程序人员提供附加信息的目的是便于生成高效的目标代码。

用过程方法实现软件系统是通过某种甚高级语言来描述问题规约、并借助其编译程序自动生成可执行的程序代码。虽然,该方法的实现效率较高,但从非算法性成分到算法性成分的转换却较难实现,甚高级语言的抽象成分越多,就越难编译实现。

过程方法的成功应用是面向领域的自动程序生成系统。其做法是,设计一种面向专门问题领域的语言及其编译程序。其基本特点是,语言为面向最终用户的描述性语言,用户只需描述做什么,无需指明如何做。这样,对于不懂程序编译技术的一般用户也可方便地使用计算机。目前,已商品化的系统不少,其中数据报表处理语言是一类典型代表。一般说来,它为用户提供若干条操作命令,用户利用这些命令,可完成报表格式定义,报表生成,查询检索数据,屏幕操作,数学运算,数据分析与通信,字处理与图形处理,以及自我教学等功能。许多工作由系统自动完成,用户编程工作量小。一般认为,只要问题领域易于规范化和模型化,就可设计出面向该领域的专用语言及其编译系统,使得对问题领域在较高层次水平上进行抽象描述。

领域知识对过程方法的自动化程度极为重要。领域知识是关于目标软件运行的现实世界的知识,包括领域术语、概念、计算公式和启发式信息等。因为领域知识是表达和理解问题需求的固有成分,所以它是任何自动程序设计系统的本质内容。领域知识能提高问题描述的抽象级别,减少总体任务的复杂性。为使那些缺少程序设计知识和编程经验的一般用户,用其所熟悉的领域术语和概念非形式地描

述待解问题,让计算机自动完成相应的程序设计任务,必须开展面向应用领域的自动程序生成系统的研究。

面向应用领域的自动程序生成系统代表性工作有下述两项。

Draco 系统是一个程序生成系统。它提供了一组设施,它可以规定以说明为主要形式的程序产生过程。当用 Draco 系统生成程序时,用户必须对使用哪些变换规则提供一定的指导。因此,Draco 系统不是完全自动化的系统。

Φ_{mix} 系统具有大量与石油测井计算有关的知识,和其他自动程序设计系统相比,它包含更多或更详细的领域知识。这类知识以规则变换的形式表示,有些规则是用过程形式嵌入系统。该系统考虑了各类知识协调使用问题。

参考文献

1. 徐家福,陈道蓄,吕建,王志坚. 软件自动化. 北京:清华大学出版社,1994
2. Barstow D. Domain-Specific Automatic Programming. IEEE Trans. Software Eng., SE-11,1985
(王志坚 张家重)

guochengshi chengxu sheji

过程式程序设计 (procedural programming) 使用过程语言设计、编写和测试问题求解程序的活动。过程式程序设计中主要涉及到数据结构的确定、求解算法的设计、代码文档的组织和测试等活动。

程序的处理对象是数据,这些数据或者是简单数据元素,或者是由一些数据元素构成的复杂数据,这些数据的逻辑结构统称为数据结构。因此,确定数据结构是过程式程序设计的首要任务。简单数据元素通常有整数、实数、布尔值、字符、枚举值和指针等,复杂数据有字符串、数组、记录、集合、链表、树和图等。为描述数据结构,程序中需要定义数据类型和变量。数据类型可分为简单数据类型和结构数据类型,分别对应于简单数据元素和复杂数据。根据变量在程序中的作用域的大小,变量可分为局部变量和全局变量。局部变量的有效范围只在程序的一小部分,全局变量则在程序的任何地方都可以进行存取。为提高程序的可读性和可修改性,程序中应该尽可能地减小变量的作用域。

求解算法的设计是指利用程序语言提供的控制结构描述求解步骤的活动。过程语言均提供顺序、

选择和循环等三种控制结构。顺序结构是指按先后顺序从前到后执行的语句序列。这些语句逻辑上可能有明确的顺序关系,即后一个程序语句依赖于前一个语句,也可能它们之间没有明确的顺序关系,即某些语句的先后顺序并不重要,一个语句逻辑上并不从属于另一些语句。编写顺序性控制结构时应该把有明确的顺序关系的语句组织在一起,对于没有明确的顺序关系的语句,根据它们对数据的依赖关系,把它们组织在一起,使代码能由上读到下,以提高程序的可读性。**选择结构**是指根据判定条件控制一些语句是否执行的语句。选择结构可用 if-then、if-then-else 或 case(或 switch)等语句进行描述。当判定条件成立时需要执行一组语句,且不成立时不需要执行这些语句,那么应当采用 if-then 语句。当判定条件成立时需要执行一组语句,否则需要执行另一组语句时,可采用 if-then-else 语句。当根据表达式的取值情况在多个动作中选取其一执行时,可采用 case 语句。**循环结构**是指可重复执行一组语句(称为循环体)的程序语句。根据重复方式的不同,循环结构可分为 while 型循环、until 型循环和 for 型循环。while 型循环是在指定的条件(称为循环条件)成立时,重复执行循环体,其特点是执行循环体前先判定循环条件,因此可能一次也不执行循环体。until 型循环将重复执行循环体,直到循环条件成立才结束该重复,其特点是每执行一次循环体后判定循环条件,因此至少执行一次循环体。for 型循环将循环体重复执行给定次数,其特点是循环开始前可确定循环次数。

人们曾提出了多种程序设计方法。自顶向下、逐步求精的程序设计方法是利用抽象机制控制程序设计复杂性的一种典型的程序设计方法。其基本思想是从最能直接反映问题体系结构的概念出发,逐步精细化、具体化、逐步补充细节,直到可用程序设计语言直接描述为止。M. A. Jackson 曾提出了基于数据结构的程序设计方法,他认为程序的控制结构依赖于程序所处理的数据的结构,这一方法给出了根据程序的输入和输出数据的结构来设计控制结构的一套方法。J. D. Warnier 的 LCP 方法是另一种基于数据结构的程序设计方法,这一方法的原理与 Jackson 方法十分相似,但 LCP 方法用更严格的逻辑方法来设计程序。面向目标的程序推导方法是一种比较形式化的程序设计方法,其基本思想是通过从程序的终结断言出发逐步推导最弱前置条件,来推导出相应的程序。

代码文档是程序的最终表现形式。代码文档的表现形式直接影响程序的可读性、可理解性。为此,应该采用好的程序设计风格和适当的注释。程序风格包括好的数据结构、好的命名方法、最小的控制流以及程序文本的清晰布局等。注释是对程序的进一步解释,注释应该描述“为什么”而不是“是什么”。

测试是为发现程序中的错误而执行程序的过程。这里程序的执行具有人工执行和计算机执行双重含义。人工执行是一人或多人发现程序中的错误,而阅读程序代码的一系列步骤和查找错误的方法,程序审查会、人工运行和复查等是常见的人工执行。计算机执行是在计算机上运行被测程序,并输入一些测试用数据,根据执行结果发现程序中错误的过程。测试方法分为黑盒法和白盒法。黑盒法着眼于程序的外部特征,而不考虑程序的内部逻辑结

构。白盒法将根据程序的内部结构,对程序的逻辑路径进行测试。需要指出的是,不管是黑盒测试,还是白盒测试,在实际运用中不可能做到完全的测试,测试只能用于发现程序中的错误,而不能用于证明程序中没有错误。为完成测试任务,需要设计测试用例,其目的在于确定一组有可能发现单个错误或一类错误的测试数据。常用的测试用例设计方法有逻辑覆盖、等价类划分、边界值分析和因果图方法等。
(金淳兆)

guocheng yuyan

过程语言 (procedural language) 参见命令式语言。

H

hanshushi chengxu sheji

函数式程序设计 (functional programming)

编写函数式程序的方法与过程。其主要任务在于定义(或构造)函数以求解所提出的问题。定义的函数(可看作主程序)可能包括一些辅助函数(可看作子程序)。计算机按照所定义函数的相应表达式,根据计算规则逐步计算,最后得到所需的结果。表达式中可能包含函数名,计算时可将其相应的定义作为归约规则。

函数式程序设计的一个显著特点是:若一个表达式有定义,则该表达式的最后结果与其计算次序无关。函数式语言可分为纯函数式语言与非纯函数式语言(参见**函数式程序设计语言**)。纯函数式语言中的变量与数学中的变量一样,它们只代表函数的参数值,在整个表达式的计算过程中其值始终不变,称为**引用透明**。基此,表达式可按照某些代数定律进行等式变换和推理。

函数式语言在表达能力方面还具有两个特点:一为构造数据的能力,即把整个数据结构看作简单值,它们可作为参数值被传送,也可作为计算结果被回送。一旦被建立后就不能改变。它们可纳入到其他结构中去,但只要需要它,其值始终有效。这使得函数式语言便于使用,但在实现时为了提高效率(特别是空间的使用效率),提出了许多研究课题。二为建立高阶函数的能力,其参数可为函数,或其结果可为函数的函数称为高阶函数。使用高阶函数可使程序简洁、清晰,但其正确实现是实现技术中的研究课题。上述两个特点构成函数式程序设计的概念架构。它既简洁、精巧,又富有灵活性和表达能力。

与使用一般程序语言相比,函数式程序设计存在如下问题:①由于函数式语言中不允许赋值语句,故无副作用,因此在表达诸如输入/输出,进程间的相互作用等方面就比较困难。虽然已有一些进展,例如使用 Monad,但离解决问题还远;②运行效率还亟待提高。究其原因,主要在于函数式程序设计语言的计算模型为输入演算,而目前的冯·诺依曼计算机是以**图灵机**为计算模型的。根本的解决途径是寻求一种适合于函数式程序设计语言的全新的

计算机体系结构。

参考文献

1. Bird R, Wadler P. Introduction to Functional Programming. Prentice Hall, 1988
2. Jones S P. The Implementation of Functional Programming Languages. Prentice Hall, 1987

(孙永强 黄林鹏)

hanshushi chengxu sheji yuyan

函数式程序设计语言 (functional programming language)

用于函数式程序设计的语言。其中函数是构造程序的基本成分,并提供一些设施用于构造更为复杂的函数。程序人员根据提出的问题去定义求解函数(即为主程序),其中可能包含一些辅助函数(即为子程序)。最早的函数式程序设计语言是 J. McCarthy 领导的小组所建立的 LISP-1 语言。1977 年, J. Backus 在他的 ACM Turing 演讲中,详细、深刻地分析了冯·诺依曼风格的语言——FP 函数式语言。他的演讲引起了世人的广泛注意,被称为函数式语言发展史上的里程碑,其后陆续出现了一些新的函数式语言如 SASL, KRL, ML, Miranda, Hope, Orwell, Haskell 等。

函数式语言可分为纯函数式语言与非纯函数式语言。纯函数式语言中禁止使用赋值语句,从而不会产生副作用,其优点是享有引用透明性,有助于程序的等式变换和推理。非纯函数式语言中,允许赋值语句在一定范围内使用,在沿袭函数式语言所享有的优点的同时,提高表达能力和实现效率。按语义区分,又可分为严格的和非严格两类。一个函数 f 称为严格的,当且仅当 $f(\perp) = \perp$ 。在严格函数式语言中,参数是按值调用,这种计算方法称为**急性**计值,反之,非严格函数式语言中,参数的计值是**惰性**的,参数只在需用时才计算,目前有许多非严格的函数式语言, P. Hudak 等人组织了一个小组,定义了 Haskell 语言,企图作为这类语言的标准文本。

参考文献

1. Bird R, Walder P. Introduction to Functional Programming. Prentice Hall, 1988

2. Jones S P. The Implementation of Functional Programming Languages. Prentice Hall, 1987

(孙永强 黄林鹏)

hanshu yu guocheng

函数与过程 (function and procedure) 低级语言中的子例程在高级语言中回送与不回送计算值的相应程序单位。二者亦可统称为子程序。这种程序单位一旦定义后,便可在一定范围内多次调用,从而避免了相同或相似的程序段在程序中多次出现。这一机制体现了计算过程的抽象。子程序有两个侧面:一个是定义计算过程的定义方面,称为子程序定义或子程序说明;另一个是使用该计算过程的调用方面,称为子程序调用。

函数与过程都是通过定义中的形式参数与调用中的实在参数传递与外部环境发生联系的,所不同的是:函数定义还需指出回送的函数值与类型,其回送值通过函数调用出现在表达式中,而过程调用往往以语句的形式出现,没有显式地立即回送计算值的功能。子程序定义中的形式参数与子程序调用中的实在参数要在个数、顺序、类型上保持一致。高级语言中的参数传递归纳起来有以下 5 种方式:

(1) 值 (ALGOL 60, SIMULA, PASCAL, Ada): 形式参数等同于子程序的一个局部变量,其初值为调用时的实参值。对形式参数的赋值不影响调用程序。

(2) 结果 (Ada): 形式参数等同于子程序的局部变量。当调用返回时,此局部变量的内容赋给相应实参,这里的实参必须是变量。

(3) 值-结果 (FORTRAN, Ada): 形式参数等同于子程序的局部变量,其初值为调用时的实参值。返回时,如果实参为变量,则把此局部变量的内容赋给此实参。实参变量在调用前被定义,或在返回前被重定义。

(4) 引用 (FORTRAN, PASCAL, Ada): 或称传地址。子程序内形式参数的所有操作均通过对其实参的引用来执行。

(5) 名 (ALGOL 60): 计算引用实参的无参过程 P 被传递到子程序,对形式参数的操作变为先调用 P,然后通过由 P 产生的引用进行操作。

在函数与过程的体部分出现了直接或间接地调用它自身时,则称此函数或过程是递归定义的。这种函数或过程在实现时开销较大,但它的表达能力强,便于实现递归算法。

当函数命名符中的实在参数变量对应函数说明中的变量形式参数,而这函数命名符的实在参数变量在表达式中作其他运算成分时;当在函数分程序内改变了不是在该函数说明内说明的变量或参数,而该改变值的变量或参数又在表达式中用作其他运算成分时,使表达式中不同计算次序产生不同的结果。这种由于函数命名符引用而引起的问题,称为**函数副作用**。

(陈涵生)

hanyu pinyin shurufa

汉语拼音输入法 (Chinese pinyin input method) 按汉语的拼音作为汉字的输入编码并输入计算机的方法。拼音方式有依照《汉语拼音方案》、依照注音字母和依照其他拼音方式等多种。这里特指按《汉语拼音方案》的字母表、声母表、韵母表及其拼写规则输入汉语拼音,然后由计算机系统自动转换为对应的汉字和词语的方法。因此,有人认为这不是一种汉字编码输入方法,而是“拼音-汉语变换法”。

汉语拼音输入法有全拼输入法和双拼输入法之分。全拼输入法按汉语音节的字母来击键输入,可以在音节之后用数字 1、2、3、4 来表示阴平、阳平、上声、去声四个声调。也可以不击调号键,只是同音字增多。双拼输入法是按音节的声母和韵母来击键输入,声、韵各一键,所以叫双拼或者叫声韵双打,零声母音节也需要加击一键表示“零声母”。双拼方案目前尚不规范。各种方案之间在声母方面主要是双字母的 zh、ch、sh 分别用哪一个键来表示不一致,在韵母方面因为要将 35 个韵母压缩到 26 个字母键的键位上,差别更大。双拼韵母在键位上的设计有依韵母使用频度和依语音“开齐合撮”规律两种不同方法,也有的设计兼顾两个方面。预计双拼设计的国家标准或行业标准会很快问世。通用键盘键入汉语拼音除了对声调如何表示、双字母声母如何表示(双拼方式时)作出规定外,还要对 ü 的表示法作出规定(通常用 yu 或 v 来表示)。此外,也有人发明声、韵、调(声调键兼空格键)三击的汉语拼音专用键盘来键入拼音,转换汉字。

拼音输入法还可以有简拼(或称拼音缩写)方式,例如 d(的 de)、wm(我们 wo3men)、jsj(计算机 ji4suan4ji1)等,一些智能化较强的拼音输入法还有“混拼”方式,即全拼与简拼的自由组合。例如,计算机(jisj;ji4suan4j;jsjil;...)。

拼音输入法的最大问题是有一些音节同音字较

多,因此,目前较流行的拼音输入法都有较大的词库支持,采用词语输入方式以减少同音字的干扰。同时也有较强的智能化手段(如自动分词、自动记忆新词、构词法分析、动态词语搭配频率统计等)以支持词语级的处理。随着智能化水平的不断提高和中、小学语文教育与普通话教育的加强,汉语拼音输入法将成为一般人使用的主要汉字键盘输入方法。

参考文献

张轴材主编. 中文信息处理技术的现状与进展. 通用中文代码国际联合会 ACCC, 1991 (张晋)

hanyu yanyu(yuyin) hecheng

汉语言语(语音)合成(speech* synthesis of Chinese) 用机械、电子、计算机等人为了的方法生成汉语言语的过程与技术。几百年前,人们就期望研制“会说话的机器”,为此研究了言语产生的机理、言语的特征、发音器官的模拟等。17世纪法国人研制了一个机械式的会说话的装置。自19世纪出现了电子合成器以后,言语合成研究得到了飞速的发展。言语合成的最终目标是让机器像人那样讲话。

1939年,贝尔实验室 H. Dudley 制作的第一个电子合成器(VODER)在美国纽约的博览会上展出。这是一个利用共振峰原理制作的言语合成器。VODER用手指按10个琴键,控制带通滤波器;3个额外的琴键控制爆破音;手腕控制声源开关;脚踏板控制张弛振荡器以改变声调,从而产生各种声音。

1960年,瑞典语言学家和言语工程学家 G. Fant 在他的论文“Acoustic Theory of Speech Production”中系统地阐述了言语产生的理论,推动了言语合成技术的进步。20世纪70年代以后,线性预测技术开始用于言语编码和识别。同时可以根据线性预测参数用多种方法来合成言语。

1980年,美国麻省理工学院教授 D. Klatt 设计的串-并联混合型共振峰合成器,用串联通道产生元音和浊辅音,用并联通道产生轻辅音,还可以对声源作各种选择和调整,以模拟不同的噪音。在此基础上开发的 DEC Talk 英语文语转换已广泛地应用于各个方面。

20世纪80年代末,E. Moulines 和 F. Charpentier 提出基于时域波形修改的言语合成算法 PSOLA,该方法较好地解决了言语拼接中的问题,从而推动了波形拼接言语合成和文语转换技术的发展和应

用。我国汉语言语合成的研究始于20世纪50年

代。80年代我国学者设计了以汉语普通话声母、韵母为合成单位的共振峰合成器。90年代初,国内的汉语 TTS 系统逐渐转向波形拼接算法。清华大学、中国科学院声学所、中国科技大学都开展了相关的研究工作。

言语合成采用的技术可分为发音参数合成、声道模型参数合成和波形编辑合成;合成策略可分为频谱逼近和波形逼近。参数合成方法通常基于言语产生的源滤波器模型,以共振峰合成为代表。它的优点是占用的存储空间小,与言语编码相结合时数码率较低,同时合成言语具有较高可懂度,能够较灵活地控制合成言语的音色,但不够自然。波形拼接方法通过对来自自然言语的言语基元进行拼接产生合成言语,具有较高的自然度。这种方法需要占用大量的存储空间,合成言语的音色相对固定。

言语合成的一个重要应用方面是文语转换(TTS),它可将文本中的文字自动转换为口语。言语合成有着广阔的应用前景,它可应用于盲人计算机、电话信息查询、文本校对、专家系统的有声输出、机场航班信息报告等领域,它也是汉语人机语音通信和智能计算机接口必不可少的组成部分。

参考文献

1. Eric Moulines and Francis Charpentier. Pitch-Synchronous Waveform Processing Techniques for Text-to-Speech Synthesis Using Diphones. Speech Communication, 1991: 453~467

2. Dennis H Klatt and Laura C Klatt. Analysis, Synthesis, and Perception of Voice Quality Variations among Female and Male Talkers. J. Acoust. Soc. Am., 1990, 87(2): 820~857

3. 蔡莲红,魏华武. 汉语文-语转换系统的研究与实现. 应用声学, 1994, 13(6) (蔡莲红)

hanyu yanyu(yuyin) lijie

汉语言语(语音)理解(Chinese speech understanding) 让计算机能理解汉语自然口语的技术。计算机不仅可以识别出连续语音的内容,而且可以像人一样理解人们说话的意思,回答人们提出的问题,或按人们的意图去执行特定的操作。自然语言理解可分为口语理解和书面语理解两方面,这里指的是口语理解。口语发音常出现习惯性

* speech 一词在许多教材和资料上也翻译为“语音”。

无意义添加词、词序颠倒、重复、停顿等现象,因此口语理解要有提取关键词的功能,这和规范的语言文字理解有较大差别。

汉语语音理解和英语、欧语相比,更为困难。首先汉语词间无间隔标记,汉语词汇自动切分本身就是一个较难的课题,由此引起的歧义现象更为严重。汉语还有连动句和兼语句两类特殊句型,再加上汉语量词丰富、成语众多,使汉语理解更难于用机器实现。基于以上原因,汉语语音理解研究目前也仅局限在极小应用范围的表演系统,而且只是发音规范的特定人连续语音识别和理解系统。例如一个词汇表为 89 个词的火车订票和信息查询连续语音理解系统,其涉及的词及由此组成的合法句子都局限在较小范围内。语音理解是人机口语对话和机器口语翻译的基础。

包括中国在内的 20 多个国家参与的国际组织“语音翻译先进研究协会”,提出了七国电话语音同声翻译研究计划(C-STAR III),是语音理解应用最宏伟的长期研究计划。

参考文献

1. Waible A, Lee K F. Readings in Speech Recognition. San Mateo: Morgan Kaufmann Publishers Inc., 1990

2. 刘开瑛,郭炳炎. 自然语言处理. 北京: 科学出版社, 1991 (莫福源)

hanyu yanyu(yuyin) shibie

汉语言语(语音)识别 (Chinese speech recognition) 由计算机系统对汉语语音所带信息进行分析与感知的过程和技术。

发展简史

语音识别的历史可以追溯到 20 世纪 50 年代。1952 年 K. H. Davis 用电阻、电容、电子管等分立元件,实现带通滤波器组进行语音频谱分析和匹配,10 个阿拉伯数字的识别率达 98%。1960 年 P. Denes 等研究成功第一个计算机语音识别系统,开创了计算机语音识别的新阶段。同年,瑞典科学家 G. Fant 提出了著名的语音产生的声源——滤波器模型,奠定了现代语音信号处理的理论基础,对语音分析、合成和识别工作起了巨大的推动作用。60 年代中期,线性预测技术引入语音信号处理,提供了较为精确的声道响应估算办法。70 年代初,动态时间规正匹配算法较好地解决了语音特征时域上非线性变化问题,提高了识别率。70 年代中期, J. K. Baker 等人

将隐马尔可夫模型(HMM)应用到语音识别领域。由于 HMM 既可描述语音的瞬态特性,又可描述语音的动态转移特性,合理地反映了语音的统计特征,在语音识别中获得极大成功,已成为当今语音识别的主流算法。近来,人工神经网络的自适应性和自学习能力对改善语音识别系统的健壮性和容错性有很大好处,受到重视。这方面研究工作集中在寻求能反映语音瞬态特性、动态特性和多变特性的神经网络,其中, Carnegie Mellon 大学 A. Waibel 在 1989 年提出的时间延迟神经网络(TDNN)是成功的例子。人对语音识别的能力是任何机器难以比拟的,特别在极低信噪比的情况下,机器识别率很差,而人耳仍能保持较高识别率。因此,在听觉机理的生理、心理研究基础上,提出了各种听觉模型,用听觉模型提取的参数用于语音识别,抗噪能力明显增强,这是语音识别不容忽视的一个重要方面。语音识别进入大词汇、连续语音识别时,首先识别声学信息,再利用语言学的知识,诸如词法、句法、语义、对话背景等知识,来帮助机器提高识别率和理解能力,特别是从已识别的语音串中挑选出正确的词句来,有重要作用。因此,语言模型的研究成为语音识别和理解的一个不可分割的后继部分(参见言语(语音)识别中的语言模型)。目前语言模型有两种,一种是用基于知识规则的方法建立模型,需要有庞大的专家知识库,句法语义分析规则十分繁复。另一种是从大量语料中统计出词搭配的概率,构成语言统计模型,利用概率分布来缩小语音识别的搜索范围和纠正误识错误。语言统计模型不但可将建立语言模型的巨大工作量由计算机负担,而且和识别模型的联接更为直接和简单,已为越来越多的语音识别系统所采用(参见汉语言语(语音)理解)。

汉语语音识别研究工作始于 1958 年,中国科学院声学研究所用电子管设备识别 10 个元音。1972 年起该所开始用计算机识别语音。20 世纪 70 年代末以来,更有一些单位,先后开展了汉语语音识别的研究,现在已有近百个单位参与这方面的研究工作。20 多年来,汉语语音识别的研究工作基本跟上了国际语音识别的研究步伐,有些地方结合汉语的特点有所创新。汉语全音节实时识别系统和某些语音文本输入系统正在向实用化方向迈进,已有一些商品问世。连续语音识别和理解正在起步,其中连呼数字串识别较为成功。强噪声下汉语有限命令的语音识别工作也取得了可喜的进展。

汉语言语(语音)的特点

汉语属于汉藏语系中的汉语语族。汉藏语系中有汉语语族、壮侗语族、苗瑶语族和藏缅语族等几十种语言。汉语语族包括多种方言。各种方言在语音、词汇、语法方面存在不同程度的差别。为使汉语表达走向规范化、标准化,国务院号召推广使用“普通话”。普通话语音音系比较简单,音节的结构形式也比较少;从听感上会觉得清亮、高扬、舒缓、柔和;在口头运用中,有鲜明的轻重音和儿化韵等变化。

汉语发音通常以一个汉字作为一个音节。语言语音体系的基本元素是元音和辅音。中国传统的音韵学研究把汉语语音分为声母和韵母。把每个音节分成两部分:字音的前一部分称为声,后一部分称作韵。辅音可以作声母,但不是所有的辅音都能做声母。元音可以做韵母,但韵母部分可以包括不止一个元音,也可以包括辅音,如 ng[N]。

汉语语音包括 64 个音素。音素构成了声母和韵母,声母(可以没有)和韵母再组合成音节。汉语有 400 多个无调音节,每个音节可以带有不同的声调,因此带调音节共 1 600 多个。

汉语是声调语言,汉语声调以其不同的调值或不同的调型区别字的意义。根据调型的差异,汉语普通话分为阴平调、阳平调、上声调和去声调,另有轻声调。当音节的声韵母相同时,可以变化声调而形成不同意义的音节,从而大大增加了汉语音节的数目,扩充了汉语语音的表达手段。事实上,汉语带调音节的数量比无调音节的数量多出几倍。如:“磨(mó)镜”(名词)和“磨(mò)刀”(动词)中的“磨”字,由于其声调不同,不但表达了不同的意义,而且连词性也不同。“北京(běi jīng)”和“背景(běi jīng)”两词的声韵母相同,仅仅是声调不同而形成了两个词。

声调是语言音高相对区别的类型,它具有修辞功能。汉语声调有抑扬起伏,高低升降的旋律性变化,使语言的表述富于形象性、生动性,增加语言的美感。

汉语的每个音节都有自己稳定的静态声调。然而,在连续语流中,受到前、后音的影响,原有的单字声调要发生变化。发生部分变调,或发生全部变调,失去原有的单字调值;或发生声调融合,即与别的声调连成一体。

基本原理

各语种语音识别的基本原理大致相同。框图如

图 1 所示。

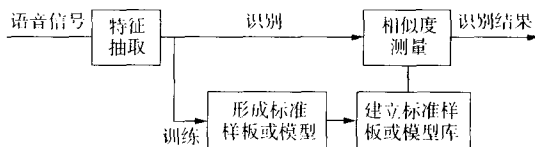


图 1 语音识别的基本原理框图

语音识别系统实现的过程分训练和识别两个阶段。训练阶段是在机器中建立被识语音的样板或模型库,或者对已存在机器中的样板或模型作特定发音人的适应性修正。在识别阶段,将被识语音的特征参量提取出来(参见语音识别的特征提取)进行模式匹配,相似度最大者(或距离测量最小者)即为被识语音。在大词汇、连续语音识别和口语理解的情况下,使用语言模型对提高识别速度和正识率起很大作用。汉语语音识别中,考虑汉语的特点会对汉语识别有所帮助。首先汉语是以音节为基础的语言,它们都是元音结尾或元音加鼻韵尾的开音节结构;汉语协同发音和音变不如英语严重;汉语是有调语言,“调”起辨意作用等。这些特点在汉语识别中可资利用。

分类

语音识别按识别对象可分为孤立词识别、连接词识别和连续语言识别与理解 3 类。按使用者适应情况,可分为认人识别和不认人识别两类(参见汉语言语(语音)识别分类)。按词表大小,可分为小词表(100 词以下)、中词表(100 ~ 1 000 词)和大词表(1 000 词以上)3 类。中、小词表识别可用整词作为识别单元。由于大词表的混淆性大大增加,只能用子词单元(音素、双音素、音节等),识别难度很大。

应用领域

语音识别的一些应用分支领域,如特定任务的口语理解系统(参见汉语语音理解)、说话人辨识和说话人确认也拥有不少研究者。说话人辨识是从一些说话人中辨认出某人来,可用于刑事侦查。说话人确认是依据说话人说出的某个特定语句,确认说话者是其本人,如在银行等系统中,用来证实确是顾客本人后才给予服务。

当前语音识别最大的问题是健壮性太差。发音人发音方式略有变化或背景噪声略有增加,就会使识别率下降,影响使用效果。这是语音识别产品难以为广大用户接受的关键问题。

参考文献

1. 陈永彬,王仁华. 语音信号处理. 合肥: 中国科学技术大学出版社, 1990

2. Waibel A, Lee K F. Readings in Speech Recognition. San Mateo: Morgan Kaufmann Publishers Inc., 1990

3. 方棣棠. 汉语语音识别的当前任务与研究方向. 见: 第三届全国人机语音通讯学术会议论文集. 成都: 四川科学技术出版社, 1994

(莫福源 方棣棠 蔡莲红)

hanyu yanyu (yuyin) shibie fenlei

汉语言语(语音)识别分类(classification of Chinese speech recognition) 对汉语语音识别系统按识别对象或按使用者适应情况进行的区分。

按识别对象有以下三类识别方式。

(1) 孤立词识别 **孤立词识别**是指在发待识别音时,单元间有明显的间歇。识别系统不需要特殊处理来分割单元。对汉语而言,以字、词或短语为单元构成词汇表,待识语音则为这些单元中的某一个。用词汇表全部或其一部分进行训练产生全部词汇的标准样板或模型(参见**汉语语音识别**)。汉语普通话中识别单元为全部可拼读音节时称全音节识别。汉语普通话实际使用的音节仅1200个左右。根据《现代汉语词典》,包括轻声在内仅有1333个不同音节,不考虑轻声仅剩1296个音节,而无调音节只有400个左右,这是汉语区别于外语的独特特点之一。全音节识别是实现汉语无限词汇识别和中文文本口呼输入的基础。

(2) 连接词识别 **连接词识别**的识别词汇表也是字、词或短语,但识别时可以是它们中间几个的慢速连读。慢速连读是指既不像孤立词识别时单元间有明显间歇,也不像连续语音识别那样需用复杂的程序来切分单元,而是机器仍很容易分割单元。识别时只会产生替代错误,不会产生插入错误和丢失错误。例如词汇表包含“0”、“1”……“9”十个数字,识别时可念“3”、“27”、“659”等。

(3) 连续语音识别 **连续语音识别**的待识语音是一些完整的句子,以正常说话速度发音,比连接词发音要快得多,甚至还允许有一定的随意性。句中每个字或词与它们单独发音比,除了有字调和词调的变化外,由于它们在句中所处位置不同,受整句语调的影响。识别单元可以字或词为单元,也可用声

母、韵母等音素为单元。由于上述原因,连续语音识别要对识别单元正确切分和识别是很困难的,是语音识别主要难题之一。在连续语音识别基础上对所识别的句子进行理解即为语音理解。

按使用者适应情况可分为以下两类。

(1) 认人语音识别 **认人语音识别**又称特定人语音识别 识别系统只适应某一个特定人。系统训练时只用使用人的语音来生成识别单元的标准样板或模型(参见**汉语语音识别**)。因而仅适合其本人使用。其他人使用时要训练产生适合自己的标准样板或模型,否则识别率将大为下降。

(2) 不认人语音识别 **不认人语音识别**又称非特定人语音识别 识别系统在不加特殊训练的情况下,可适应某一范畴的说话人(例如说普通话的人),而不是仅适应某个特定人。这种系统显然需要由一些属于这一范畴的发音人来训练标准样板或模型。该系统可供参加训练者(圈内人)使用,也可供未参加训练者(圈外人)使用。由于不同讲话者之间不仅发声器官(声道长度、鼻腔大小等)有差异,而且发音习惯如口音、速度、响度等也有所不同,因而严重影响识别效果,使不认人识别也成为语音识别难题之一。不认人识别可用两种自适应方法来提高识别率:一种方法是将发音人分为几类,每一类包含类似的发音人,并有其特定的标准样板或模型。识别之前发音人先发一句或几句训练句,由识别系统进行分类后将相应标准样板或模型调入识别器中。另一种方法是用一个经过良好训练的非特定人或上述特定分类的标准样板或模型,发音人发少量自适应训练句,对已存入的标准样板或模型参数进行修改,使之适应该特定人。

参考文献

1. 中国社会科学院语言研究所词典编辑室. 现代汉语词典. 北京: 商务印书馆, 1996

2. Waibel A, Lee K-F. Readings in speech recognition. San Mateo: Morgan Kaufmann Publishers Inc., 1990

(莫福源 方棣棠)

hanzi

汉字(Hanzi, Chinese character, Han character, Chinese Hanzi) 主要用于记录汉语(我国台湾地区习称国语,新加坡、马来西亚等地亦称华语)的文字。

汉字属于表意文字。相对于表音文字(或拼音文字)而言,表意文字是用符号直接表达词或词素。

在信息的国际标准中,往往用“表意文字(ideographic character)”泛指汉字,而用 Chinese Hanzhi/Japanese Kanji/Korean Hanja 特指在中国、日本、韩国所使用的汉字。

我国汉字自古至今累计总数已超过 70 000 字,但现代汉语常用字的数量远少于此。国家语言文字工作委员会颁布的《现代汉语通用字表》包含 7 000 字。

在信息处理中,汉字的字形、字音、字义及由此派生的一些属性起着十分重要的作用。

汉字的字形属性 汉字字形是汉字形体结构的图像,是汉字的表现形式。除了极个别的特例,汉字都是方块型的结构。汉字字形属性包括以下内容:

(1) 部首 汉字形体偏旁所分的门类。我国以 201 部首为现代汉语部首的推荐规范。国际上习用《康熙字典》的 214 部首。在对汉字排序时,如果部首是排序的因素,则每一个汉字的部首必须惟一。

(2) 部首外笔画数 一般与部首联合使用,用于排序和(或)检索。

(3) 总笔画数。

(4) 结构特征 指一个汉字形体的左右结构、上下结构、包围结构和镶嵌结构等。

(5) 汉字笔画序列 按照汉字的规范书写顺序,将汉字拆分为横竖点撇折(一丨丶丿乙)等笔画的序列。

汉字的字音属性 汉字字音源于汉语读音。除了极个别的特例,一个汉字只有一个音节。汉字读音由 400 多个无调音节构成。加上四声变化,共有 1290 多个带调音节。汉字具有较强的多字同音性质。

(1) 拼音带调 指汉语拼音,一般用小写字母,四声用 1,2,3,4 表示,5 表轻声。

(2) 注音带调 指“国语”(我国台湾地区称汉语为国语)注音,第一声不标,第二、三、四声用 ˊˋˋˋ表示,· 表轻声。

汉字的字音属性可以用于汉字的排序,同音字再按照笔画的数目排序。汉字的字音属性与其字义密切相连(与词语背景相关),在言语识别、言语合成等过程中起着重要的作用。

汉字的字义属性 每个汉字的含义一般常有 2~5 个,有的多达 6~9 个,少数字甚至有十几种不同的意义。汉字的字义需根据它们的上下文进行分析才能确定。汉字又具有很强的构词能力,2 个或多个汉字可以构成以千万计的不同含义的词汇、名词和术语,这就给汉字文档的自动分类、自动摘要、关

键字提取、文档检索、自动翻译等计算机处理带来了许多困难。

汉字还有一些其他的属性,如汉字的使用频度、构词力(一个汉字与其他汉字一起构成复词的能力)、关联属性(如汉字的简体、繁体,正体、异体,新、旧字形之间的相互关系)等,它们在汉字信息处理中分别起着不同的作用。

参考文献

中国社会科学院语言研究所词典编辑室. 现代汉语词典(2002 年增补本). 北京:商务印书馆, 2002
(张轴材 张福炎)

hanzi bihua shurufa

汉字笔画输入法 (Chinese character stroke input method) 采用选定的汉字基本笔画作为码元来输入汉字的方法。汉字笔画输入法是汉字字形输入法之一。基本笔画多选定横(一)、竖(丨)、撇(丿)、点(丶)、折(乙)五种,有的将折又分为左折和右折两种(或称为弯和折),共 6 种基本笔画,也有选 8 种或更多种作为基本笔画的。还有的在基本笔画之外又增加一两种高频的部件(字根),如:口、十等,输入时以笔画为主。

笔画输入法码元少,易学易记。其编码规则需要首先进行笔画认同,例如:挑(乚)入横,捺(㇏)入点,竖钩(亅)入竖或入折等等。其次要规定编码的顺序,有的依笔画顺序编码,有的依字形笔画高低由高向低编码,有的依字形部件顺序取一、二、末三个部件的头二笔或头三笔码相加等等,各种编码原则都分别有一批字有歧义。由于码元少,为了将重码率降低到一定程度,笔画编码的平均码长一般较长。有的笔画输入法采取一键双笔画的方法,例如:将横、竖、撇、点、折两两相配成为横横(一一)、横竖(一丨)、横撇(一丿)、横点(一丶)、横折(一乙)、竖横(丨一)、竖竖(丨丨)、竖撇(丨丿)……共计 25 组双笔画。

笔画输入法的码元可以定义在通用键盘的数字键上,也可以定义在字母键上,还可以另外造专用小键盘输入。当只用数字键输入或在专用小键盘输入时,笔画输入法可以单手操作录入汉字,这对于某些特殊环境(如:舰艇)或特殊人员(如:残疾人)有重要意义。在掌上型的电脑大规模推入市场后,这种输入法也有可能显示出其键元少的巨大优越性。

参考文献

1. 陈一凡,胡宣华. 汉字键盘输入技术与理论

基础. 北京: 清华大学出版社, 1994

2. 张普. 汉字键盘输入方法. 语文建设, 1992, 3
(张普)

hanzi bianma (jianpan) shuru fangfa

汉字编码(键盘)输入方法 (Chinese character coding (keyboard) input method) 采用某种汉字编码方案, 通过键盘由操作者向计算机输入汉字的方法。

从不同的角度, 可以对汉字编码(键盘)输入方法进行不同的分类。从操作者的角度看, 可以有普及型和专业型两大类。普及型输入法面向一般的使用者, 这些人一般采用“想打”(依据腹稿, 边想边打)方式, 不需要追求高速录入, 但希望方法易学、易用, 不易忘。普及型输入法应该适应人民群众已有的语言、文字背景知识, 这样可以基本上免去对操作者的编码培训。当前我国人民群众的语言、文字背景知识可以定位在初中毕业水平。专业型输入法是录入员、打字员、发报员等专业人员使用的, 这些人一般采用“看打”(有书面文稿, 边看边打)、“听打”(有语言文稿, 边听边打)两种方式, 他们一般追求速度, 可以接受职业培训。但从长远看, 专业型录入人员最终是不必要的。国家应该选择普及型输入法作为在全国推广使用的汉字输入法。

从汉字特征信息编码的角度看, 可以分为音码输入法、形码输入法、音形(或形音)码输入法(参见**汉字音形输入法**、**汉字形音输入法**)。音码输入法主要以汉语拼音输入法为主, 分为全拼和双拼两大类型, 也有采用注音字母和另外发明的拼音文字作为代码的。形码输入法以汉字部件(字根)输入法为主, 分为音托和形托两大类型。也有以笔画方法编码输入的(参见**汉字笔画输入法**)。一些形码输入法还辅以字形结构信息或完全依靠字形结构信息来编码。字形结构信息包括部件的数量及部件与部件的结合方式等。

从键盘的角度看, 可分为使用专用键盘的输入法和使用国际通用 ASCII 键盘的输入法。另外还有一种所谓“无键盘汉字输入法”, 实际上是把键盘移到屏幕上, 用鼠标、光笔、手指来点触。其中, 在移动计算设备(如个人数字助理 PDA, 手机等)上使用小键盘方便地输入汉字的方法已经受到普遍关注。

20 世纪 70 年代末期到 80 年代初期的汉字编码(键盘)输入方法多为单纯字输入方式, 可称为字处理阶段; 80 年代中期, 计算机中汉语词库建立, 出

现字词联想、词语联想、词为基础字为辅助等多种输入方式, 可称为词处理阶段; 80 年代末期开始出现一些更新的智能化手段, 采用词性、词法、词语搭配频率、句法甚至部分语义和语用知识来输入汉字, 输入法更加方便用户, 拥有自动分词、自动进行前后缀匹配、自动记忆新词、自动记忆动态搭配频率、自动进行词组或句子转换等智能技术, 可称为语句处理阶段。今后将主要在规范化和智能化两个方面继续研究和提高, 使汉字编码(键盘)输入方法的通用性更强, 更适应计算机硬、软件环境的发展变化。

随着语言、文字研究的不断深化和信息处理用语言、文字标准的陆续颁布, 加上其他相关硬、软件标准和规范的实施, 汉字键盘输入方法在规范化方面会越来越突出, 同一类型的输入法将逐步接近。汉字键盘输入的平台技术已经推出, 将在规范化和智能化方面发挥重要作用。大规模的精加工语料库的开发将使计算机获得越来越丰富的语言知识甚至背景知识。汉字键盘输入方法在智能化方面也会越来越强。基于汉字信息库的键盘输入方法已进入市场, 不同类型的输入法将可能共享大型汉语知识库(包括汉字、语音、词汇、句法、语义、语用等多方面的知识信息)。

参考文献

1. 陈一凡, 胡宣华. 汉字键盘输入技术与理论基础. 北京: 清华大学出版社, 1994
2. 张普. 汉语信息处理研究. 汉字键盘输入方法及其评测研究. 北京: 北京语言学院出版社, 1992
(张普)

hanzi bianma zifuji

汉字编码字符集 (Coded Chinese Character set (狭义), Coded Ideographic Character Set (广义)) 按照一组无歧义的规则而定义的汉字字汇的集合, 其中每一个汉字都有其惟一的代码表示。汉字编码字符集用于汉字信息的表示、处理、交换、传输与存储。

我国汉字的总数超过 6 万, 数量大, 字形复杂, 同音字多, 异体字多, 我国的香港、澳门和台湾地区, 以及日本、韩国、新加坡、马来西亚等也都使用汉字。因此, 如何确定字符集中收入多少字、哪些字, 它们的排序方式, 编码空间的安排以及编码方法, 与其他编码字符集的关系等是一个非常复杂的问题。

目前在计算机信息系统中使用的汉字编码字符

集有多种,它们大体可以分成两类:一类是以汉字字汇作为主体的汉字编码字符集,另一类是不仅包含汉字字汇而且包含世界各国和地区使用的主要文字符号的多文种编码字符集。前一类大多依托 ISO/IEC 2022 所定义的体系结构,后一类则采用 UCS/Unicode 所定义的体系结构。

以汉字字汇作为主体的汉字编码字符集 ISO/IEC 2022 定义的编码字符集体系结构,采用单字节的 256 个码位的代码空间。扣除控制字符占用的 64 个码位,图形字符可使用的代码空间就比较狭小。因此,不同国家和地区对使用的文字符号只能分别进行编码,即使是欧洲共同体也使用好几种不同的编码字符集来涵盖所使用的文字符号。汉字是大字符集,每个汉字至少需要用 2 个字节来表示。中、日、韩等国家和地区,对使用的汉字分别进行编码,它们的字汇、字级和字序各不相同。此类汉字编码字符集在东亚地区最主要的有如下几种。

(1)《信息交换用汉字编码字符集·基本集》(GB 2312—1980) 这是 1981 年我国颁布的第一个汉字编码国家标准,该标准选出 6 763 个常用汉字字符和 682 个非汉字字符,为每个字符规定了标准代码。其中一级常用汉字 3 755 个,二级常用汉字 3 008 个。

(2)《信息交换用汉字编码字符集·辅助集》(GB 12345—1990) 这是为了使用繁体汉字而特地制定的繁体汉字字符集的国家标准,它与 GB 2312 相对应,含 6 866 个汉字。其中简、繁体相同的汉字,字型与编码均保持不变;简体汉字对应 1 个繁体字的(如简体字“灯”的繁体为“燈”),编码不变但字形替换为对应的繁体字;一个简体汉字对应多个繁体字的(如简体字“发”,对应“發”、“髮”2 个繁体字),原简体字替换为最常用的那个繁体字,其他对应的繁体字放在扩充区,这样的简体字有 300 多个。

(3)《汉字内码扩展规范》(GBK) 这是我国 1995 年颁布的汉字编码的指导性规范,它与国家标准 GB 2312—80 信息处理交换码所对应的事实上的内码标准兼容,共有 21 003 个汉字和 883 个图形符号,在字汇一级支持 ISO/IEC 10646—1 的全部中日韩汉字(CJK)共 20902 字。

(4)《台湾地区标准汉字字符集》(CNS 11643—1992) 其全称为《通用汉字交换码》。共收入汉字 13 053 个(不使用简化汉字)。与 CNS 11643—1992 对应的内码为 Big 5 码(俗称“大五码”),通常都用 Big 5 泛指二者。Big 5 码与 GB

2312 的内码不兼容,需要进行转换才能正确地显示与打印汉字。

(5)《日本工业标准汉字字符集》(JIS X 0208—1990) 其全称为《情报交换用汉字符号系》。共收入汉字 6 355 个。其中一级字(即第一水准)2 965 个,按假名顺序排列;二级字(即第二水准)3 390 个,按部首、笔画数排列。

(6)《韩国国家标准汉字字符集》(KSC 5601—1987) 其全称为《情报交换用字符集》。共收入汉字 4 888 个,其中有 268 个同音重见字,按韩文读音排序;还有韩文数千个。

包含汉字字汇的多文种编码字符集 不同国家和地区对使用的字符集分别进行编码会产生许多问题。例如,编码系统会互相冲突,两种编码字符集可能使用相同的代码代表两个不同的字符,或使用不同的代码代表相同的字符;任何一台计算机(特别是服务器)都需要支持许多不同的编码字符集;数据在不同的系统之间交换时,总会有损坏的危险。

解决上述问题的方案是采用统一编码,即不论什么计算平台,不论什么程序语言,世界各国和地区使用的所有文字符号都采用一个惟一的代码。UCS(通用多八位编码字符集)和 Unicode 定义的字符集编码体系结构就是为此目标而开发的。UCS 和 Unicode 两者完全兼容,其体系结构基于所谓“多八位”编码(4 字节或 2 字节)。目前在工业上实现的均为双字节的 UCS-2 形式,即所有字符都集中在一个平面(共 65 536 个码位)内。UCS/Unicode 编码的体系结构的特点是:编码空间大,能容纳世界上所有国家和地区使用的文字和符号;编码空间连续,不用像 ISO/IEC 2022 体系那样躲避与控制字符相关的大片区域;按文字编码而不是按语言、按国度编码,中日韩统一汉字编码即是在这样的原则下进行的;中、英文等长编码,便于计算机处理;字符的编码既用作交换码,又可以用作内码(处理码),改变了 ISO/IEC 2022 体系结构中汉字交换码与内码不一致的情况。以 UCS/Unicode 作为体系结构的包含汉字字汇的多文种编码字符集有如下几种:

(1) ISO/IEC 10646—1:2000(Unicode 3.0)和 ISO/IEC 10646—2:2001(Unicode 3.1)标准 ISO/IEC 10646(Unicode)编码字符集中的汉字,是遵守中、日、韩汉字认同甄别规则而得到的,包括 CJK 汉字(20 902 个)、CJK 扩充 A(6 582 个)和 CJK 扩充 B(42 778 个)。它们源自中国及其台湾地区、日本、韩国的 13 个字符集,它涵盖了 GB 2312 之全部,

GB 8565 之全部,《现代汉语通用字表》之全部, Big 5 之全部,中国台湾地区新、旧电报码之全部,日本的 JIS 汉字之全部,韩国 KSC 汉字之全部。

(2) GB 13000—1:1995 GB 13000 是等效采用 ISO/IEC 10646 的中国国家标准版本。目前 GB 13000 是 1995 年版,对应 ISO/IEC 10646—1:1993(第 1 版)。日本的对应版本是 JIS 0221:1995,韩国的对应版本是 KSC 5000:1995。

(3) GB 18030—2000(《信息交换用汉字编码字符集·扩充集》) UCS/Unicode 编码中的汉字及其编码与我国已使用多年的 GB 2312 和 GBK 标准并不兼容,为了既能尽快地向 ISO/IEC 10646(Unicode)编码标准过渡,又能向下兼容 GB 2312 和 GBK 汉字编码标准,因而制定了 GB 18030—2000 汉字编码国家标准,并在 2001 年开始执行。GB 18030—2000 采用单字节、双字节和四字节编码,码位总数达 160 多万个,能完全映射国际标准 UCS/Unicode 的基本平面和辅助平面中的字符集。它包含的汉字数目增加到 27 000 多个,同时还收录了藏文、蒙文、维吾尔文等主要的少数民族文字,可适应出版、邮政、户政、金融、地理信息系统等领域的用字问题。考虑到我国用户的需要及解决现有系统的兼容性和对多种操作系统的支持,采用 GB 18030 是我国目前汉字编码的较好选择。

参考文献

<http://www.unihan.com.cn/cjk/cjkhome.htm>

(张福炎)

hanzi bujian(zigen) shurufa

汉字部件(字根)输入法(Chinese character component input method)

选取若干有代表性的汉字的部件(又称字根)作为码元,以汉字字形结构及其切分规则作为编码的主要依据,确定每个汉字的输入代码的方法。汉字部件(字根)是由笔画组成的具有组配汉字功能的构字单位。这种输入法是汉字形码输入法之一。汉字部件(字根)输入法的方案有等长码(每个汉字编码的码元数相同)和不等长码两种设计,采用中键盘或小键盘两种方式的键盘。中键盘一般是专门设计的,通常每键上定义一个或一组字根,有的设计还考虑了字根在键盘上的位置与字根在字中的固定位置相对应,如“彳,亻”在左、“刂,攴”在右、“艹,艹”在上、“灬,讠”在下等。小键盘一般采用国际通用键盘,每键有一组字根,字根的分组定位或依形似原则(可称

形托),例如:将“日、白、目、自、四、血”放在 B 键上,或依音似(常为声母)原则(可称音托),例如:将“木、马、目、母、皿”放在 M 键上。有些还考虑了组字频度或使用频度,使每键负担趋于平均,以降低重码。此外,也有人考虑利用每个汉字的字根数、字根位置及汉字结构类型来给汉字编码。

目前,国家尚未公布汉字的部件集或基本部件集的标准,中、小学也还没有统一的部件教学规范,不同的汉字部件(字根)输入法之间大同小异,其区别主要在于:选取的部件数不同,部件的分组归并原则不同,部件定义的键位不同,汉字的切分规则和切分下限不同,单字和词语的取码规则不同,简体字形与繁体字形的处理方式不同等。

规范化是汉字部件(字根)输入法今后发展的主要研究方向之一,上述各种不同的方面均需要进行规范,这种规范要兼顾到汉字的造字法和现代汉字字形的演变,兼顾到汉字的部件教学法和已有的汉字部首检字法,以使规范具有科学性,继承性,大众性,这样才易学,易用,便于推广普及。

参考文献

1. 傅永和. 汉字结构和构造成分的基础研究. 见: 现代汉语用字信息分析. 上海: 上海教育出版社, 1993
2. 傅永和. 汉字结构及其构成成分的统计及分析. 见: 现代汉语定量分析. 上海: 上海教育出版社, 1989
3. 张普. 汉字部件分析的方法和理论. 见: 语言自动处理. 武汉: 武汉大学出版社, 1988 (张普)

hanzi jianpan shuru

汉字键盘输入(Chinese character input via keyboard)

操作者通过键盘向计算机等信息设备手动键入汉字的过程、技术和方法。又称为汉字编码输入。它是计算机和其他信息设备(如手机、PDA、电子词典等)输入汉字的主要方法之一。

向计算机输入汉字的方法有两类。一类是自动识别输入,包括汉字的自动识别(参见汉字识别)和汉语语言的自动识别(参见汉语语言(语音)识别);另一类就是汉字键盘输入。由于计算机最早由西方国家研制开发,它使用的键盘是面向输入西文字符设计的,一个或两个西文字符对应着键盘上的一个按键。汉字是大字符集,国家标准汉字编码字符集包含的汉字已达 2 万多字,专用的一键一字的汉字输入键盘由于键太多、查找不便、成本又高等原因早

已不再采用。利用只有几十个键的计算机键盘(甚至只有十几个键的手机按键)输入汉字时,无法使每个汉字与键盘上的按键一一对应,因此必须用一个或几个按键的组合来表示汉字,这就是汉字的键盘输入编码。

设计一种汉字键盘输入编码方案,首先要利用汉字的音、形等特征信息,按照一定规则,对指定的汉字编码字符集中的每一个汉字进行描述,然后再确定这些特征信息与键盘按键之间的对应关系,这样就可以在普通西文键盘上输入汉字了。

汉字的键盘输入编码方案有几百种之多,能够被广泛接受的编码方案应具有下列特点:易学习、易记忆、效率高(平均击键次数较少)、重码少、容量大(可输入的汉字字数多)等。到目前为止,能够在所有方面都做得很好的编码方法还不多。

汉字输入编码的方案可以从不同的角度进行分类。例如从使用者的角度看有普及型(面向一般用户)和专业型(面向专业的数据录入人员)两类。从编码特征的角度看大体可以分成4类:①数字编码。这是使用一串数字来表示汉字的编码方法,例如电报码、区位码等,它们难以记忆,不易推广。②字音编码。这是一种基于汉语拼音的编码方法,简单易学,适合于非专业人员;缺点是同音字引起的重码多,需增加选择操作。③字形编码。这是将汉字的字形分解归类而给出的编码方法,重码少,输入速度较快,但编码规则不易掌握,五笔字形法和表形码属于这一类。④音和形结合的音形码或形音码。它吸取了字音编码和字形编码的优点,使编码规则适当简化、重码减少,但掌握起来也不容易。

近年来,在上述编码输入方法的基础上,利用计算机的高速处理和存储能力,充分发挥计算机的统计学习功能,实现字词联想、词语联想,并采用词性、词法、词语搭配频率、句法甚至部分语义和语用知识来输入汉字,同时还自动记忆新词,自动调整词语频率等,这些所谓的“智能汉字输入法”,受到了广大用户的欢迎。

汉字键盘输入的编码方案虽然很多,但其中有些不符合国家语言文字的规范。例如,各种各样的汉字拆分方式正在万“码”奔腾地大比拼,虽然都能解决汉字的计算机输入问题,但它们对汉字的不规范和无序拆分已经使汉字文化受到污染、干扰和破坏,贻害无穷。为此,我国从20世纪90年代中期开始加快了语言文字和中文信息处理领域的立法和国家标准、规范的制定。已经公布的与汉字键盘输入

有关的国家标准主要有:

GB/T 18031《信息技术数字键盘汉字输入通用要求》

GB 15834《标点符号用法》

国家语言文字委员会颁布的规范有:

GF3001《信息处理 GB 13000—1 字符汉字部件规范》

GF3002《GB 13000—1 字符集汉字笔顺规范》

GF3003《信息处理用汉语拼音方案表示规范通用键盘》。

另外,还有即将公布的 GB XXXXX《信息技术通用键盘汉字输入通用要求》。

上述标准与规范将纳入国家技术法规而强制执行。作为产品出售的汉字键盘输入系统,均应遵循上述标准和规范,并将编码层次和软件层次视为统一的汉字键盘输入系统进行性能考核和产品认证。

国家标准 GB 18031 和 GB XXXXX《信息技术通用键盘汉字输入通用要求》中关于汉字键盘输入系统的性能指标有3个:易学性、汉字输入平均码长和重码字词键选率。易学性指的是学会使用汉字编码输入系统的时间应尽量短,并应符合使用汉语作为母语的使用者的思维习惯;汉字输入平均码长的定义是:在输入给定的测试样本时,测得的输入每个汉字的平均击键次数,计算公式为:平均码长=输入样本的击键次数/测试样本总字数(键/字);重码字词键选率的定义是在输入给定测试样本过程中,通过重码选择键确认的汉字字数与测试样本总字数的百分比,计算公式为:重码字词键选率=(重码选择键确认的字数/测试样本总字数)×100%。

作为面向市场的汉字键盘输入系统,应该通过标准符合性测试、产品论证和专家技术鉴定。其中,标准、规范、性能指标测试由政府授权的中文信息处理产品标准符合性检测中心进行,具有一票否决权。

参考文献

1. 张普等. 论汉字键盘输入技术——历史·现状·展望. 见: 汉字编码键盘输入文集. 北京: 中国标准出版社, 1997
2. <http://hzdt.xiloo.com/index.html> (张福炎)

hanzi neima kuozhan guifan—GBK

汉字内码扩展规范——GBK (Chinese Internal Code Extension Specification, GBK) 由电子部和国家技术监督局于1995年12月颁布的,

大大扩充了 GB 2312 字汇的汉字编码字符集的指导性规范 (GBK 的 K 是“扩展”的汉语拼音第一个字母)。

GB 2312 只有 6 763 个汉字,在人名、地名的处理上经常令人十分尴尬,尤其是在古籍整理、古典文献研究方面有很大缺憾。为此,迫切需要有包含更多汉字的标准字符集。

GBK 字符集中一共有 21 003 个汉字和 883 个图形符号,它与 GB 2312 国标汉字字符集对应的事实上的内码标准兼容。除了 GB 2312 中的全部汉字 (称为 GBK/2) 和符号 (称为 GBK/1) 之外,还收录

了包括繁体字在内的大量汉字 (GBK/4 与 GBK/3) 和符号 (GBK/5),例如“計算機係”等繁体汉字和“冂冂冂有銘”等生僻的汉字,在字汇一级支持 ISO/IEC 10646—1 和 GB 13000—1 的全部 20 902 个中日韩 (CJK) 汉字。

GBK 字符集中的每一个字符都采用双字节表示,总的编码范围为 8140~FEFE,首字节在 81 与 FE 之间,尾字节在 40 与 FE 之间 (剔除 xx7F 一条线不安排字符),总计 23 940 个码位,共收入 21 886 个汉字和图形符号,未使用的区域作为用户自定义区 (图 1)。

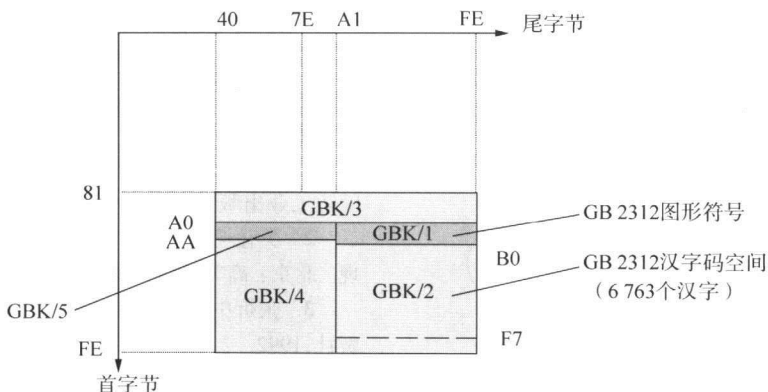


图 1 GBK 汉字在双字节代码空间中的码位

自 Windows 95 和 Windows 98 中文版开始,微软公司 Windows 操作系统的几乎所有中文版都已经全面支持 GBK。

参考文献

<http://www.uniha.com.cn/cjk/cjkhome.htm>

(张福炎)

hanzi shibie

汉字识别 (Chinese character recognition)

利用计算机将书写或印刷的汉字图像自动转换为表示该汉字的计算机代码 (例如汉字国标码、Unicode 码等,参见汉字编码字符集) 的过程、方法和技术。印刷体汉字识别可以将大批印刷文字材料自动高速输入计算机,在数字图书馆、电子出版、数据库建设等领域有重要的应用。联机手写汉字识别是便携式信息设备输入汉字的主要手段之一。

汉字识别基本上是一种模式识别,它还涉及图像处理、人工智能、统计决策理论、信息论、中文信息处理等学科,也涉及视觉心理学、语言文字学等。汉

字识别过程包括汉字图像信号获取 (印刷或手写书面文字利用扫描仪的扫描图像输入,联机手写汉字利用书写板传感器的书写笔迹获取)、汉字的分割、汉字特征提取和选择、基于汉字特征的汉字分类判决,以及利用上下文对分类判决结果的验证处理等。由于汉字的数量极大 (数千乃至数万字)、结构复杂、字形千变万化,在相当长的时间内,汉字识别是最困难的文字识别问题,也是最困难的模式识别问题之一。

光学字符识别 (OCR) 最早起源于 1929 年, Taushek 在德国获得了一项有关 OCR 的专利。欧美国家从 20 世纪 50 年代就开始了西文 OCR 技术的研究,以便代替人工键盘输入浩如烟海的文字材料。汉字识别的研究是 1966 年开始的,由于方块汉字完全不同于英文的文字,汉字识别的研究受到人们的特殊重视,“汉字识别”也因此成为重要的技术名词。

1966 年 IBM 公司的 Casey 和 Nagy 发表了第一篇关于印刷体汉字识别的论文。20 世纪 70 年代以

来,日本学者做了许多工作,其中有代表性的系统是1977年东芝综合研究所研制的可以识别2 000个汉字的单体印刷汉字识别系统;80年代中期,东芝、松下、理光和富士等公司的基于专用硬件的印刷体日文识别系统走向市场。我国自70年代末开始了对汉字识别的研究,到80年代末研究开发成功印刷汉字识别系统和联机手写汉字识别系统的初级产品。在90年代,印刷体汉字识别和联机手写汉字识别的实用化产品逐渐完善,90年代末开始走向成熟,汉字识别已在国民经济的许多领域得到广泛应用。

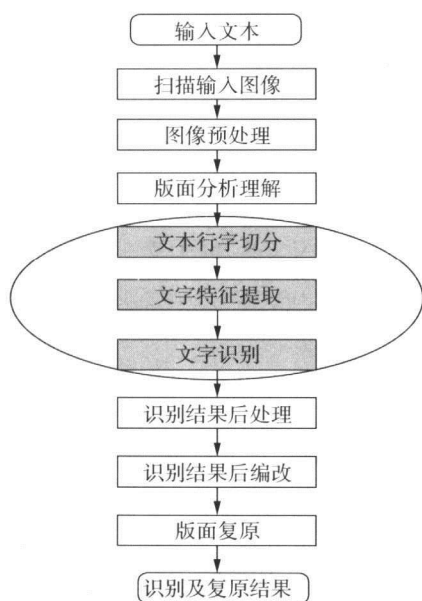


图1 印刷体汉字识别的基本流程

(注:圆圈内是汉字识别的核心部分)

汉字识别的任务是将成千上万种汉字字符图像类别进行辨识,属于超多类模式识别问题。汉字识别按照汉字输入方式的不同,有脱机识别和联机识别两种;按汉字生成方式的不同,有印刷体识别和手写识别两种。由此,汉字识别基本上可分为印刷体汉字识别、联机手写汉字识别、脱机手写汉字识别3种。

联机手写汉字识别 由于手写笔在图形书写板上写字的同时,能将每个汉字的笔画轨迹位置以及书写笔画顺序信息实时输入计算机,因此计算机可以利用获得的笔画轨迹和书写顺序识别汉字,大大减轻了联机手写汉字识别的困难。

脱机汉字的识别 无论是印刷汉字识别或脱机

手写汉字识别,没有汉字的笔画和笔顺信息可直接利用,所获得的仅仅是经过扫描仪扫描获得的汉字的图像信息。对汉字的图像进行模式识别比联机汉字识别更加困难。其中,因为印刷体汉字的字形有相当的一致和规整性(仅仅因字体不同而字形有所变化),使同一文字的差异和变化较小,识别的困难有所下降。而脱机手写汉字识别则由于不同人的书写风格和习惯不同,使手写的同一汉字的字形迥异,给手写汉字识别带来巨大困难。

经过20多年的努力,我国目前在印刷汉字识别、联机手写汉字识别和规则书写的脱机手写汉字识别方面,均已经开发出高性能可实用的系统。它们广泛应用在各种领域中。即使是最困难的自由书写汉字识别也已取得了有限程度使用的进展。

参考文献

1. 郭平欣,张淞芝. 汉字信息处理技术. 北京:国防工业出版社,1985
2. 吴佑寿,丁晓青. 汉字识别原理、方法与实现. 北京:高等教育出版社,1992
3. 张忻中. 汉字识别技术. 北京:清华大学出版社,1992 (丁晓青)

hanzi shibie houchuli

汉字识别后处理 (Chinese characters recognition postprocessing) 利用文本的语言模型(句法和语义规则)对汉字识别的结果加以限制,以减少识别结果的不确定性,提高文本识别率所进行的计算机处理。

文本作为书面语言,其内容是受到上下文限制的。利用这种上下文语言的约束就可以对文本识别的结果进行改进。

由汉字识别分类系统获得的识别结果是一个按一定置信度递减顺序排列的候选字符串,例如最小距离分类系统获得的识别结果是按距离从小到大顺序排列的候选字符串(参见汉字识别基本方法),在不考虑上下文语言模型的情况下,将识别结果的候选字符串中最小距离(或最大可信度)者,作为最终的识别结果;在考虑上下文的情况下,从识别候选字符串中选择使整个词或句子得到最优结果的作为最终识别的结果。

考虑上下文识别的后处理方法基本上有两种:一种是利用词条的前后联想匹配方法,将与词条不符的识别结果加以修正;另一种是基于语料库的统

计语言模型方法。

上下文处理的基于语料库的统计的语言模型的方法是将基于统计的马尔可夫语言模型应用于文字识别。即假定语言是一个 $n-1$ 阶的马尔可夫链,即当前符号的出现仅与它前面的 $n-1$ 个符号有关,而与更前面的符号无关,称这种语言模型为 n 元文法模型(n -gram 语言模型)。利用大规模的语料库进行加工、统计,可以得到 n 元文法模型的同现概率和条件概率等参数。

在利用 n -gram 语言模型进行上下文后处理时,把具有确定性边界的一个语言序列(一般为一个句子)作为一个处理单元,用动态规划方法从单字识别给出的识别候选字符序列中找出构成概率最大的一个句子,从而可以在相当程度上纠正识别时仅以单字识别存在的错误。

n -gram 语言模型,可以是基于字的,也可以是基于词的,还有基于词类的 n -gram 等等。

基于字的 n -gram 模型是利用相邻字之间的约束关系,从候选字符串中确定最佳的识别结果,而不需要汉语分词处理。常用的字 bigram 模型是利用相邻 2 个字间的约束关系的,由于常用汉字数目一定,基于字的 bigram 模型参数容易获得,数据存储空间不大,处理速度快而且易于实用。trigram 模型则考虑了相邻 3 个字间的约束关系,比 bigram 的相关约束更可靠,但所需数据存储空间大,处理速度较慢。

基于词的 n -gram 模型是利用相邻词间的约束关系,由于词是最小的独立语言单位,基于词的 n -gram 模型显然应当比基于字的 n -gram 模型更为有效。但是,必须基于汉语分词处理,而且词的数量远超过字的数量,给模型的训练带来巨大困难。

利用语料库统计的上下文处理方法可以自动纠正句子中连续几个字的错误,大大超过靠词条的前后联想匹配方法,对于识别率的提高有较明显的作用,得到了广泛的应用。

(丁晓青)

hanzi shibie jiben fangfa

汉字识别基本方法 (basic method of Chinese character recognition) 应用模式识别或模式分类原理对输入计算机的汉字图像提取特征并进行识别分类的方法。

汉字识别的方法有结构识别和统计识别两种。结构识别方法是在提取汉字笔画的基本结构基元的基础上,对汉字结构进行匹配而识别的。但是,由于提取有限汉字结构基元的不稳定性,结构识别方法

在解决实际汉字识别问题时未见成效。而利用汉字图像的高维统计特征,按照统计模式识别方法对汉字分类识别,则由于高维统计模式识别的强鲁棒性特点,使得即使在实用中汉字受到强烈干扰和字形发生巨大变化的条件下,也能较好解决超大字符集汉字的识别问题。

汉字的笔画信息是汉字的本质特征。在统计识别中利用汉字总体结构下笔画的“微结构”形成的统计量,是一种比直接抽取笔画结构更鲁棒和更有效的利用结构信息的方法。

在统计模式识别中,将提取的汉字样本特征用 N 维向量 $X = (x_1, x_2, \dots, x_N)^T$ 表示。汉字字符的模式类别数集合为 $\omega_i \in \Omega = (\omega_1, \dots, \omega_C)$, 类别数为 C 。在学习阶段,利用已知类别的样本进行学习,获得字符分类的鉴别函数 $g_i(X)$, $i = 1, \dots, C$; 在识别阶段,则利用鉴别函数 $g_i(X)$ 对未知类别的待识样本 X 的类别属性进行判决。即

$$\omega(X) = \omega_i \quad \text{当 } g_i(X) > g_j(X)$$

$$\text{对所有 } j \neq i, \quad i, j = 1, \dots, C$$

如果在学习阶段利用训练样本获得特征的概率分布,识别时按照贝叶斯最大后验概率准则进行判决,即判决 $\omega(X) = \omega_i \in \Omega$, 满足

$$\omega_i = \arg \max_i P(\omega_i | X)$$

将获得最小错误概率的分类,这是设计最优分类器的目标。

在简单近似的情况下,利用学习样本获得该类模式特征的平均向量 M_i , 对于待识样本 X , 利用最小距离判决函数,即 $g_i(X) = \|X - M_i\|^2$, 获得最小距离分类器判决,即 $\omega(X) = \omega_i \in \Omega$, 满足

$$\omega_i = \arg \min_i \|X - M_i\|$$

最小距离分类器识别方法简单易行,在某些情况下也可获得相当好的识别性能,从而得到较普遍使用。

在整个汉字集合的识别系统中,对于每个汉字类别的鉴别函数的参数数据集合构成了汉字识别的所谓“识别字典”。例如,在应用最小距离分类系统时,“识别字典”中包括的是被识别汉字集合中每一汉字类别的特征向量均值的总和。

参考文献

吴佑寿,丁晓青. 汉字识别原理、方法与实现. 北京: 高等教育出版社, 1992

(丁晓青)

hanzi shibie tezhen

汉字识别特征 (features for Chinese charac-

ter recognition) 为识别汉字而对汉字图像进行分析所得到的表征汉字字形结构规律的关键性的、稳定的某种结构、参数或它们的相关数学表示形式。

汉字识别中,除了分类判决的鉴别函数外(参见**汉字识别基本方法**),最重要的是汉字的特征抽取和选择。应当说,特征选择决定了识别分类系统的鉴别性能,而鉴别实施的性能则由分类系统决定。识别特征的选择以能提供足够的有效互信息熵为准,以满足识别的要求。在特征与识别模式间的有效互信息熵 I 表示为

$$I(E, F) = I(F, E) = - \sum_{i=1}^C \int_{R^N} P(X, \omega_i) \log_2 \frac{P(X, \omega_i)}{P(\omega_i)P(X)} dX$$

式中: E 为模式类别空间, ω_i 为模式类别向量, C 为类别数; F 为特征空间, X 为特征向量。 $P(X, \omega_i)$ 为联合概率密度函数。

有效互信息熵准则说明,一个好的识别特征必须满足:①与识别类别密切相关;②特征对类内变化比较稳定,受到噪声和字形变化影响较小。因此,汉字识别的研究过程,也包括对特征选择的不断发展,以及对汉字图像识别的特征选择问题的进一步认识。

汉字识别的特征有粗外围特征、粗网格特征、复杂指数和四边码、笔画密度特征、汉字特征点、边框和局部特征、部件模板、笔画方向和轮廓特征、网格单元、笔画序列和各种数学变换特征,等等。这些特征在识别汉字时各有特色,互有优劣,曾被应用于各种识别方案的粗、细分类。汉字识别研究发展至今,人们认识到汉字识别特征抽取的方法可以有多种,其中基于汉字的整体图像获取充分的信息最重要,例如,基于方向线索的网格特征取得了比较优秀的识别效果。

为了获得优良鉴别能力的识别特征,在原始抽取的特征上进行特征的变换,也成为提高汉字识别性能的有效手段。例如,对原始特征向量进行 PCA 主分量分析,可以进行维数压缩,减少学习样本不足产生的维数危机问题;利用 LDA 线性鉴别分析,以获取最有鉴别能力的特征分量等。这些利用特征变换使特征鉴别能力优化的方法,都将有效地提高和改进汉字识别系统的识别能力。

参考文献

吴佑寿,丁晓青. 汉字识别原理、方法与实现. 北京: 高等教育出版社, 1992

(丁晓青)

hanzi xingyin shurufa

汉字形音输入法 (Chinese character calligraphic and phonological input method)

采用汉字字形及字音两种属性作为码元的一种汉字编码(键盘)输入法。它以字形属性为主,字音属性为辅。例如:采用字根方式给汉字编码之后(参见**汉字部件(字根)输入法**),为了区分重码字,再增加一个整字的汉语拼音第一个字母,比如,想:木+目+心+x, x 即 xiang 的首字母;又如,部:立+口+β+b, b 即 bu 的首字母。至于前面的字根输入设计是和汉字部件(字根)输入法的介绍相一致的。也有的拆取汉字的字根后再加上其他字形信息,以整字、字根和其他信息的读音首字母编码。例如加上整字的字型信息,通常把字型分为左右字(□□)、上下字(□□)、包围字(□□)、独体字(□)四类,可以用每一类首字声母编码,如 Z 代表左右字, S 代表上下字, B 代表包围字, D 代表独体字。又例如,在《见字识码》方案中,前三码采用部件和整字的读音首字母,第四码采用末笔笔画读音首字母,如:张 = G(弓) + C(长) + Z(张) + D(末笔“捺”入“点”)。

参考文献

支秉彝. 见字识码法. 见: 汉字编码方案汇编. 北京: 科学技术文献出版社, 1979 (张普)

hanzi yinxing shurufa

汉字音形输入法 (Chinese character phonological and calligraphic input method)

采用汉字字音及字形两种属性作为码元的一种汉字编码(键盘)输入法。它以字音属性为主,字形属性为辅。例如:采用汉语拼音的声、韵、调三种字音属性为主给汉字编码之后,为了区分大批的同音字,增加一个部首(汉字字形属性)信息为辅。he2 可以是“和、何、河、荷、合…”很多同音字的编码,但 he2 + 禾部、he2 + 亻部、he2 + 彳部、he2 + 艹部、he2 + 人部……则上述同音字区分开了,它们分别是“和、何、河、荷、合……”等字的编码。部首的表示可以按音托或形托两种不同的方式来定义(参见**汉字部件(字根)输入法**)。

音形输入法还可以有其他设计方法。例如:辅助的字形属性可以是部件(或部首)、可以是笔画、可以是字形结构等,为主的字音属性可以是全拼式的,可以是双拼式的(参见**汉语拼音输入法**)等等,但都是以字音属性为主字形属性为辅的方法。也有

的方法采用等长码四码来编码,前两码是双拼的声码,后两码是两个表示部首或部件的代码,双声双部,难以说是哪方面为主哪方面为辅。这种方法本质上是汉语拼音输入法和汉字形码输入法相结合的产物。汉字音形输入法优点是避免了全拼音的码太长和同音字太多,又不使字形信息的取码太复杂,缺点是同时要掌握音、形两种取码方法或规则,对一些用户来说也不是太方便。(张普)

hanzi zihui

汉字字汇 (Ideographic repertoire) 用汉字编码字符集或类别指定的汉字集合。用编码字符集指定的字汇,如:《信息交换用汉字编码字符集·基本集》(GB 2312)用字;日本工业标准汉字字符集(JIS 0208)用字。用类别指定的字汇,如:简化字,繁体字,传承字,异体字等等。字汇本身不涉及编码的概念,只是表明“多少字”、“哪些字”。

(1) 简化字 又称简体字,是笔画简化了的汉字,用来代替原来通行而笔画较繁的汉字。如“劉”简化为“刘”,“滅”简化为“灭”,“馬”简化为“马”等等。通常所称的简化汉字是指 1986 年 10 月 10 日国家语言文字工作委员会经国务院批准重新发表的、原中国文字改革委员会于 1964 年编印的《简化字总表》中的简化汉字,总数为 2 235 个字。

(2) 繁体字 简化字的对称,已有简化字代替的笔画较多的汉字。如“禮”是“礼”的繁体字。GB 2312 中的简体字所对应的繁体字有 2 000 多个。

(3) 传承字 在文字发展过程中,未经整理简化的汉字亦即不存在简体、繁体之分的汉字。如:一、二、三、天、地、人、上、中、下、南、北等等。GB 2312 中的传承字有 4 000 多个,占整个字符集汉字的 2/3。

在实际生活中,当谈到繁体字或规范字时,往往指的是语境或集合。比如,所谓“繁体字”,实际上是指繁体字和传承字的集合;当谈到规范字时,是指简化字加上传承字。

(4) 异体字 跟规定的正体字同音同义而写法不同的字。如“攷”是“考”的异体字。有的异体字常常被认为是繁体字,如“災”、“廂”、“決”等字常常被视为“灾”、“厢”、“决”的繁体字。

(5) 类推简化字 按照《简化字总表》的第二表【可作简化偏旁用的简化字和简化偏旁】,用整体简化的字和偏旁类推出来的简化汉字。有的在《简化字总表》中已经列举,有的还没有列举。

(6) 略字 日本简化汉字。如“学”、“对”、“疗”、“团”等等。

(7) 外字 日本对编码字符集之外的汉字的俗称。

在使用“外字”一词时,指明编码字符集是至关重要的。比如,一个日文 JIS 的外字,就很可能是中文 GB 2312 字符集的“内字”;一个 GBK 的内字,又很可能是 GB 2312 的外字。

参考文献

中华人民共和国国家标准. 信息交换用汉字编码字符集基本集 GB 2312. 北京: 中国标准出版社, 1980 (张轴材)

hanzi zixing

汉字字型 (Hanzi font, Hanzi typeface) 用同一方法、同一风格制作的汉字造型的集合。同一字型的所有字符的造型数据也称为字库,在电子出版领域字型也称为字模。

字型的分类 在计算机中,根据不同的造型方法将字型分为点阵字型、矢量字型和轮廓字型。

(1) 点阵字型 用 $m \times n$ 的像素阵列表示字符形状的造型方法所产生的字型。其优点是便于显示器、打印机、照排机等光栅扫描输出设备的输出。但占用的存储空间大,进行缩放变换时,字型质量难于保证。目前,我国已制定了国标一级和二级汉字的 16×16 , 24×24 , 32×32 的点阵字型标准。

(2) 矢量字型 用一组折线表示字符形状的造型方法所产生的字型。其优点是便于笔式绘图机等随机扫描输出设备的输出,所占用存储空间小,但较之用曲线描述,在有些情况下不够美观。

(3) 轮廓字型 当字符笔画的内外轮廓线均用一组直线段或曲线段来描述时,称为轮廓字型。其优点是字型质量高,占用的存储空间小,便于无级变倍和字形变化。目前,所有的视窗平台都采用这种字型。

字型的风格 具有同一风格的字型称为一种字体。常用的汉字字体有数十种,如宋体、仿宋、黑体、楷体、圆体、魏碑、隶书、行书等。

计算机中使用的汉字字型有许多种,出版、广告等领域使用的汉字字型多达几十甚至上百种。在设计制作汉字字型时,应注意汉字笔形的美观、笔画的粗细、字的重心、字的骨架等,务必做到一副字型和諧、协调,体现汉字的整体美。(张轴材 付艳丽)

hanzi zixu

汉字字序 (Hanzi order) 汉字编码字符集中汉字排列的顺序。这种排列顺序,依汉字编码字符集的不同,有很大的差异。例如,在基于 ISO 2022 的汉字编码字符集和基于 ISO /IEC 10646 和 Unicode 的汉字编码字符集中,汉字字序有:

(1) GB-2312 中汉字的字序 其一级字是以汉字的汉语拼音的拉丁字母序排列的。同音字以笔形顺序:横、竖、撇、点、折为序。起笔相同,则按第二笔。依此类推。而在二级字中,汉字主要是以其部首及部首外的笔画数排列的。笔画数相同,则按笔形顺序(横、竖、撇、点、折)为序。起笔相同,则按第二笔,依此类推。

(2) Big 5 中汉字的字序 Big 5 包含常用字和次常用字两个字面的汉字。每个字面中,汉字主要是按其总笔画数从小到大排列的,笔画数相同时,再按部首顺序排列。

(3) JIS x 0208 中汉字的字序 其一级字是按日文的假名读音;二级字是按部首及部首外的笔画数排序的。

(4) KS C 5601 中汉字的字序 韩文汉字是按其韩文读音排序的。

(5) ISO /IEC 10646 和 Unicode 中汉字的字序 在该字符集中,汉字是按照同一个汉字在中国、日本、韩国四大字典(康熙字典、大汉和辞典、汉语大字典、大宇源)中的页码、字位、序号综合排序的。

由于同一个汉字在不同的语言中有不同的读音,标记读音的方法也各不相同,所以,在国际标准中,汉字不可能按音排序。同时,由于各个国家和地区公认的部首数、部首序不同,每个字的写法的不同又会带来笔画计数的差异,所以,在国际标准中,汉字也不可能简单地按部首排序。(张轴材)

hedai qudongqi

盒带驱动器 (cartridge tape drive) 驱动并控制盒式磁带的传动以实现数据的写入与读出的设备。是一种以盒式磁带为存储媒体的存储设备。磁带封装在带盒中,使用时连同带盒一起装到驱动器内。

盒带驱动器多按流式工作,记录段与记录段间在记录时不需要启停动作。它省却了在开盘式磁带驱动器中为了快启停动作所必需的低惯量大转矩伺服电机及其大功率驱动放大电路、带的缓冲机构以

及真空泵等部件,因而带的传动机构简单,且功耗小,噪声小,体形小,性能可靠。

盒带驱动器一般都具有自动装带与卸带功能。当带盒插入驱动器的加载窗口时,盒被自动拖进驱动器。到位后,盒上的存取门被打开,带被自动地安排在与磁头及主动轮正常工作的接触位置。卸带时,带首先脱离与磁头及主动轮的接触,退回到盒内带盘上,盒的存取门关闭,盒被退出。IBM 3480 型单带盘盒带的情形有些特殊。当带盒到位后,带端塑料导引块被穿到驱动器的带盘上。卸带时,全部磁带需退回到带盒,带端导引块脱离机器带盘,锁回到带盒上,带盒被退出驱动器。

由于厂商各自发展技术,盒带有许多种格式,因而盒带驱动器的类型也很多。按所采用的记录方式可分为线性纵向记录、螺旋扫描记录与横向扫描记录等三种类型。

20 世纪 80 年代初期产生的线性纵向记录磁带技术利用宽阔的磁带记录面,可以获得更大的存储面积,通过增加磁道数量来提高存储容量。各种线性纵向记录磁带技术的记录方法相类似,不同之处在于单位记录密度、编码方式、磁道数量上的差异。线性记录磁带机械结构简单、精度高,磁带的磨损低,有利于保护磁带中的数据。设备主要类型有 IBM 3480 型、QIC 型、DLT 型、SLR 型、SDLT 型、LTO 型等。

20 世纪 80 年代后期发展起来的螺旋扫描记录技术是源于数字化视频与音频记录技术,工作原理与家庭录像机相似。由镶嵌在高速旋转鼓面上的磁头在慢速走动的磁带上按斜线轨迹进行扫描记录。磁带需部分地围绕在鼓面上,鼓的轴线和带前进方向的垂线有一很小的斜角,以便在带上扫描出斜线段的磁道来。不同类型的驱动器所用的扫描鼓的直径、鼓的转速、带的速度、读写头的路数各不相同,道密度和位密度也都有不小差异。有些类型的设备还用了正负方位角记录的方法,即每两相邻道上对应磁头磁化缝隙偏斜的角度方向刚好相反,而相邻道的磁化图形如鱼骨状,这样可以减小从邻道读到的干扰成分,相邻磁道可更紧密地贴在一起,从而显著提高了道密度。

螺旋扫描记录驱动器的基本结构都很相似。由于带速很低,因而走带机构简单。头鼓倾斜安装在底座上,装带时由两个导柱将带从盒中拉出,使带包在鼓面上。卸带时导柱缩回,供带盘将带收回到盒内,盒被退出驱动器。

螺旋扫描记录的设备主要类型有 4 mm 数字音频盒带 (DAT)、8 mm 视频盒带、AIT 格式盒带等类型。

横向扫描记录盒带驱动器系仪表数字记录用设备,在计算机领域很少应用。

在同一类型的盒带驱动器中,因容量(磁道数量、位记录密度、带长以及是否采用数据压缩技术)与带速的不同有多种型号。此外,按磁头同时读写的路数又有并行与串行之分。除用于大型计算机的 IBM 3480 型盒带驱动器为并行方式外,其他类型的盒带驱动器基本上都是串行方式。在串行方式中,多数设备采用单磁头读写,也有一些设备为了适当改善数据传输速率而采用双磁头(或四磁头)同时读写的串行方式。在采用纵向记录的串行读写方式驱动器中,磁头需横跨磁带移动,以便更换磁道。通常称这种横向的磁头移动为蛇行。

小型盒带驱动器还习惯从设备外形尺寸上区分,比如有 5.25 英寸与 3.5 英寸等规格以及全高度设备与半高度设备之分。采用数据压缩技术的驱动器可将盒带的正常容量增加一倍以上。盒带驱动器是用于数据备份最主要的设备。

多年来,磁带上磁介质的性能也有显著的提高。其材料从早期的氧化铁涂层发展到高级金属蒸发层,因而单位记录密度获得了大幅度提高。磁带的寿命也提高到 30 年以上。

3480 型盒带 IBM 公司于 1984 年推出的磁带存储子系统。它是传统的开盘磁带机的替代机型。它的出现标志着磁带存储技术的重大变革。驱动器首次用上了薄膜磁变阻磁头与小型盒带,磁带运动不用记录块间快启停方式而代之以流式。磁带从盘到盘直接传动,由两带盘测速器及带的张力传感器输出的信号送至数字信号处理器,对带速与张力进行伺服控制。设备结构较以往开盘式快启停的磁带机简化了许多,设备体积显著减小。

3480 型盒带采用二氧化铬记录介质,带长 170 m,连同单个带盘封装在 125 mm × 109 mm × 25 mm 的塑料盒中。在半英寸磁带上记录 18 条磁道,位记录密度为 750 b/mm,数据传输速率为 3 MB/s,一盒磁带的容量为 200 MB。3490 型盒带驱动器具有正反向读写的功能,磁道数增加一倍,容量为 400 MB。3590 型盒带有 256 磁道,容量达 40 GB。

QIC 盒带 是一种常见的小型盒带。QIC 盒带技术是 20 世纪 80 年代初由 3M 公司开发的。它采用多磁头部件,在直线磁道上串行读写。主要特点

在于使用了结构非常精密的盒带,而驱动器本身的机构却很简单。故 QIC 盒带较其他类型的盒带贵很多,而设备却较其他类型的盒带驱动器便宜。QIC 盒带驱动器为了便于在微型计算机上安装,采用了与磁盘驱动器相同的外形尺寸。大型 QIC 为 5.25 英寸,有全高度的和半高度的两种。小型 QIC 为 3.5 英寸。1982 年的 QIC 24,容量为 60 MB,磁道数为 9,位记录密度为 315 b/mm(8 000 bpi)。

QIC 盒带的磁带宽度为 6.35 mm(0.25 in)。有两种带盒尺寸,大的是 101.6 mm × 152.4 mm × 16.9 mm,小的是 61.3 mm × 81.0 mm × 14.5 mm。

QIC 中的 Travan 技术是 3M 公司于 1995 年推出的。2003 年,Travan 40 达到非压缩容量为 40 GB,数据传输速率为 4.0 MB/s。Travan 磁带机的特点是价格低、生产厂商多、兼容性好。主要用于 PC 机的备份。

DLT 格式盒带 由 DEC 公司于 1985 年开发。DLT 以纵向曲线型记录法在磁带上进行记录,具有大容量、高速度的特点。它使用单轴 1/2 英寸磁带盒。容量从 2.6 GB 到 160 GB 不等,非压缩数据传输速率为 0.8 MB/s 至 10 MB/s。

SDLT 格式盒带 Quantum 公司 2001 年推出的格式,它在 DLT 技术基础上结合新型磁带记录技术,采用高级金属粉末(AMP)作为磁带的记录介质,并使用激光导引磁记录(LGMR)技术增加磁带表面的记录磁道数,使记录容量增加。2003 年,SDLT 磁带的单盒容量达到 300 GB,数据传输速率为 36 MB/s。SDLT 对 DLT 有良好的向下兼容性。

SLR 格式盒带 由 Tandberg 公司开发。驱动器仅有两个运动部件(磁头和驱动电机),磁带不需要被拉出磁带盒,对磁带磨损小。SLR 格式盒带驱动器采用了低成本的双通道边写边读磁头组件,可以在数据备份的同时进行读检验。磁带装卸检索速度快是其突出的优点。2002 年开发的 SLR-7 容量达到 20 GB,数据传输速率为 6 MB/s。

LTO 格式盒带 由 HP、IBM 和 Seagate 联合开发。它类似于 DLT,采用线性多通道双向磁带格式、硬件数据压缩和高效率纠错技术,以提高磁带的容量和性能。LTO 有 Ultrium 和 Accellis 两种格式。前者的特点是大容量,后者的特点在于存取速度快。常见的是 Ultrium 格式的产品,采用单轴 1/2 英寸磁带。2003 年,其非压缩存储容量达 100 GB,数据传输速率达 20 MB/s。

4 mm 数字音频盒带 由 HP 和 Sony 公司开发。

原是用干记录数字音频信号,用螺旋扫描技术将信号转化为数字后存储。用作计算机存储器时仍沿用原来的磁道记录格式,而文件记录格式则有 DDS 和 DATA/DAT 两种。DDS 格式的驱动器按流式方式工作,有 DDS-1, DDS-2, DDS-3, DDS-4 各种规格。DDS-1 具有 1.3 GB 的存储容量和 182 KB/s 的数据传输速率。DDS-4 单盘有 40 GB 的存储容量和 6 MB/s 的数据传输速率。

4 mm 数字音频盒带的外形尺寸为 73 mm × 54 mm × 10.5 mm,盒内装 3.81 mm 宽的金属粉末磁带,长度有 60 m 至 150 m 不等的规格。4 mm 数字音频盒带是盒式磁带中最小巧的。由于磁带宽度太窄,可记录面积有限,进一步提高存储容量有一定困难。

8 mm 视频盒带 Exabyte 公司 1987 年开发。采用螺旋扫描技术,其特点是磁带容量大,传输速率高,早期驱动器以 EXB 8200 型产品为代表,鼓的直径为 40 mm,转速为 1800 r/min,带速为 10.9 mm/s。道密度为 32 道/mm,位密度为 17 000 b/mm,一盒 110 m 长的磁带可存储 2.33 GB 数据。2003 年推出的 Mammoth-2,一盒 225 m 长的磁带可存储 170 GB,数据传输速率达 30 MB/s,(压缩后)。8 mm 视频盒带的外形尺寸为 95 mm × 62.5 mm × 15 mm,磁带宽约 8 mm,是金属粉末磁带。

AIT 格式盒带 Sony 公司开发。AIT 磁带驱动器中应用了大量新技术,如高级金属蒸镀带(AME)技术、螺旋扫描技术、磁带盒内存储器(MIC)技术。AME 磁带的记录密度比传统的金属粉末(MP)磁带高 1 倍,而且不会污染磁头,因此不需要定期清洗磁头。采用录像机中所用的螺旋扫描技术使 AIT 磁带机具有更高的记录密度、更高的可靠性。MIC 技术提高了磁带的寿命,同时使数据搜寻速度快,且磁带装卸时间也短。2001 年面市的 AIT-3 磁带未压缩容量为 100 GB,数据传输速率为 12 MB/s。

(刘锡刚 林兼)

hong chuli chengxu

宏处理程序 (macroprocessor) 把源程序中的宏指令或宏语句扩展成等价的、预先定义的指令序列或语句序列的处理系统。

宏指令或宏语句实际上是按规定格式书写的某一源程序段的缩写。它们通常是用户根据自己的特定需要,采用程序设计语言所提供的指令或语句来定义,称之为**宏定义**,其中应给出宏的名字、格式、参数和等价的指令序列或语句序列。对于常用的宏指

令或宏语句亦可由系统预先定义,供用户直接引用。当用户在程序中要使用宏指令或宏语句功能时,只要按宏定义的格式,给出宏的名字及其相应的参数即可,这称之为**宏调用**。当宏处理程序将源程序中出现的宏调用扩展成等价的宏指令序列或宏语句序列时,称之为**宏扩展**。建立宏处理程序后,用户可以方便地定义和使用自己所需的宏指令或宏语句。这不仅能简化应用程序的编写,而且有助于软件人员研究和移植有关的软件。例如,利用宏指令或宏语句设计虚拟机,研究新的语言,以及生成带有变化成分的软件等。

宏处理程序通常采用两遍算法实现:第一遍收集宏定义的信息;第二遍对源程序中的宏调用实施宏扩展。在第一遍扫描中,遇到宏定义时,应把名字、格式、参数等信息以及随后的等价的指令序列或语句序列记录到宏定义表中。对于源程序中宏定义以外的部分,将不加改变地复写到目标程序区中。第二遍扫描第一遍所产生的中间结果程序。遇到宏调用时,则将宏定义表中相应的等价指令序列或语句序列复写到目标程序区中。复写过程中,要用宏调用中的实在参数替换宏定义中的形式参数。如果限制每个宏调用只能调用前面已定义的宏指令或宏语句,那么这种宏处理程序的实现算法可合并成一遍算法来完成。

功能较强的宏处理程序还可增加嵌套宏定义、嵌套宏调用或条件宏处理等功能。如果宏定义中含有另外的宏定义,则称为**嵌套宏定义**。如果宏定义 A 中出现宏调用,那么在扩展 A 的宏调用过程中,又要进一步转去扩展其他的宏调用,这种情况称为**嵌套宏调用**。如果宏处理程序能根据宏调用中的特殊参数,有选择地把宏调用扩展成不同的指令序列或语句序列,则称为**条件宏处理**。

参考文献

1. Campbell-Kelly M. An Introduction to Macro. London: MacDonald, 1973
2. Brown P J. Macro Processor and Techniques for Portable Software. London: Wiley, 1974
3. Cole A J. Macro Processor. Cambridge: Cambridge Univ. Pr., 1976

(曹东启)

hongmo shibie

虹膜识别 (iris recognition) 计算机利用人的虹膜纹理信息自动识别其身份的过程和技术。虹膜是位于眼睛黑色瞳孔和白色巩膜之间的圆环状部

分,呈现一种由内向外的放射状结构,包含有许多相互交错的类似于斑点、细丝、冠状、条纹、隐窝等形状的细微特征,如图1中圆环内部区域所示。以上这些细微特征称为虹膜的纹理信息。它们主要由胚胎的发育环境差异所决定,对每个人来说都是惟一的,因而可以用来进行个人身份的认证。测试表明,虹膜识别通过合适的算法其准确性是所有生物特征识别中最高的。

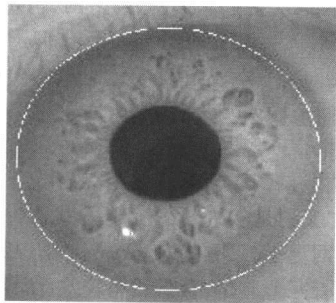


图1 虹膜图像(参考文献1)

虹膜识别的思想可以追溯到20世纪30年代。1936年,眼科专家 Frank Burch 指出虹膜具有独特的信息,可以用于身份的辨识,但是并没有引起广泛的注意。后来直至1987年才由医生 Leonard Flom 和 Aran Safir 提出了利用虹膜图像进行自动识别的概念并获得专利。1991年美国的 Johnson 实现了第一个自动虹膜识别系统。1993年 John Daugman 研制成功高性能的虹膜识别算法,得到大家的认可并仿效。其后虹膜识别的研究得到飞速发展。

目前虹膜识别系统在国外已经日趋成熟,并且已经在金融、公安、国防等领域得到初步应用。而国内从事虹膜识别的研究相对较晚,1998年中科院自动化所开始了虹膜识别的相关研究,并已取得了初步成果。

虹膜识别主要包括虹膜的检测与定位、特征的提取以及识别等。虹膜检测从输入的原始图像中确定虹膜与巩膜的外边界以及虹膜与瞳孔的内边界,以提取其中的虹膜部分作为后续工作的基础。而特征的提取则从虹膜图像中选择所需要的特征用于识别。

特征的提取是虹膜识别的关键。目前国际上比较有影响的方法是 Daugman 于1993年提出的基于相位分析的方法,该方法采用 Gabor 小波滤波的方法编码虹膜的相位特征,然后利用归一化的 Hamming 距离实现特征匹配。其基础是 Gabor 小波具有和人类简单视觉细胞相似的视觉特征,可以很好地分析现实世界中的各种模式。基于相位分析的方法

是目前识别性能最好的方法。1998年 Boles 等人提出了基于过零点检测的方法。该方法采用一维小波对沿虹膜中心同心圆的一条采样曲线进行虹膜中纹理的过零点检测,将其作为虹膜的特征,并进行识别分类。另外一类比较常用的方法是基于纹理分析的方法,比如可以采用 Gabor 滤波器提取虹膜在不同频率和方向下的纹理信息,或者采用 Haar 小波对虹膜图像进行分解提取其中的高频信息作为特征等。

2001年英国国家物理实验室的测试表明,虹膜识别通过合适的识别算法,其准确性是所有生物特征识别中最高的。因此,在一些安全性能要求较高的领域,虹膜识别具有广阔的应用前景。目前虹膜识别的主要难点在于鲁棒的特征描述和提取方法。由于瞳孔随着光照条件的变化而引起虹膜的形变,眼球的旋转、睫毛对于有效虹膜区域的遮挡等,这些都会影响到特征提取算法的性能。虹膜识别的另外一个难点在于活体虹膜的检测,也是目前研究人员关注的焦点。

参考文献

1. John G. Daugman. Biometric Personal Identification System Based on Iris Analysis. United States Patent, No. 5291560, 1994
2. Wildes R P, Asmuth J C, et al. A Machine-vision System for Iris Recognition. Machine Vision and Applications, 1996, 9: 1 ~ 8
3. Boles W W, Boashah B. A Human Identification Technique Using Images of the Iris and Wavelet Transform. IEEE Trans. On Signal Processing, 1998, 46: 1185 ~ 1188

(吴志勇 蔡莲红)

hulian wang

互联网 (interconnection network, ICN)

通过直接或间接方式将一个并行计算机系统各个处理单元(也称为结点)或多个处理机、存储模块以及各种外部设备相互连接起来,在系统软件控制下,相互通信和协调工作的硬件网络结构,它是并行计算机系统重要的组成部分。

并行计算机系统内部的互联网与连接多个计算机系统的局域网或广域网在地理分布、传输速度与控制方式等方面有区别。因此,互联网这一术语一般只在讨论并行计算机内部通信时使用。但是,随着通信技术和分布式计算技术的进步,过去用于远程通信的装置也逐渐用来作为松散耦合并行计算机系统或机群系统的通信部件,互联网与一般计算机

网络的界限将越来越模糊。

发展简史

在并行计算机出现以前,电话通信已经广泛使用。20世纪五六十年代电话通信事业的迅速发展,推动了开关连接系统的基本理论研究,其成果集中反映在 C. Clos 关于非阻塞开关网络的论文和 V. Benes 关于连接网络和电话通讯数学理论的经典著作中。并行计算机互联网中非阻塞 Clos 网和可重构 Benes 网即来源于这些成果。互联网中常用的交叉开关也是采用电话通信中的术语。互联网最先在美国伊利诺依大学研制的 ILLIAC IV 阵列机中采用,这种网络当时主要用于处理机单元之间的数据置换,故当时称为置换网络。美国卡内基-梅隆大学于 1974 年研制成功的包括 16 台处理机的多处理机系统 C. mmp 最早采用交叉开关实现存储器共享。美国 BBN 公司 1989 年推出的 TC-2000 率先在商品化并行计算机中采用多级互联网。Intel 公司和 NCUBE 公司在其推出的大规模并行计算机中分别采用的网格与超立方体互联网是点到点静态互联网的代表。

分类与结构参数

互联网可分为静态与动态两大类。在静态互联网中,作为通信用的开关元件和缓冲寄存器等分散设置在每一个结点内,各结点之间的连接在设计机器时已经固定,程序运行时的连接关系不能动态设置。静态互联网中结点间的连线是无源的,其连接通路称为被动连接通路。这种互联网应和应用问题的通信模式匹配,不同的静态互连结构适合于不同的应用。静态互联网的主要优点是可伸缩性强,成本较低。已生产的大规模并行计算机普遍采用这种互联网。与此相反,动态互联网通过控制信号控制网络中的开关元件与仲裁部件,按程序运行的要求建立系统中各功能部件之间的连接通路。因此,处理机之间或各功能部件之间的连接可动态改变。这种互联网具有较强的通用性,对应用问题的通信模式限制较少。但相对于静态互联网,其成本较高,技术较复杂,可伸缩性较差。大多数通用的多处理机系统都采用动态互联网。

描述一个互联网通常采用结点度、网络直径、对分宽度以及对称性等参数。在表示网络拓扑结构的图中,结点所连的边数称为该结点的度。结点度反映每个结点的输入输出接口数,因而直接与网络成本有关。网络中任意两结点间沿最短路径通信所经过的边数称为两结点的距离,网络中任意两结点间距离的最大值称为网络的直径。网络直径与通信延

迟有一定的关系。如果将一互联网分成相等的两半,在各种对分方法中,连接这两半的最小连接边数称为网络的**对分宽度**。一个好的互联网应当通信能力强、成本低而且可伸缩性好,这就要求结点度小而且一致、网络直径小而且随结点数增加只缓慢增加、对分宽度适中而且网络对称性好、不含通信瓶颈。

静态互联网的拓扑结构 静态互联网可以有各种不同的拓扑结构,较常见的包括线形网、环形网、带弦环形网、树形网、星形网、网格网、超立方体网和全连接网等。一维的线形网最简单,但网络直径最长,从一端向另一端发送信息要经过网中所有结点。环形网将线形网两端连接起来,所以网络直径减少一半。如果将环形网上各结点的度从 2 增加到 3 或更大,就构成带弦环形网。在结点度为 3 的静态互联网中,二叉树是较理想的互连方式。树形网的直径比环形网更小,只有 $2 \log_2 N$ (N 是结点数)。但此网络中没有冗余通路,容错性能较差,而且系统中信息流量不均匀,根结点附近的结点可能成为通信瓶颈。这一缺点在**胖树网**中得以克服。在胖树网中,从叶结点到根结点,通信带宽逐步增加。星形网是一种两层的高结点度树形网。网格网是将邻近结点按规则的平面几何图形连接起来的互联网。矩形连接的网格网是最常用的网络结构之一。早期的并行计算机 ILLIAC IV, DAP 和后来的大规模并行计算机 CM-2, Paragon 都采用网格网。纯粹的网格网是不对称的,边界结点的度数少于内部结点。为了减少网络直径,可将网络周围的结点对接起来形成筒形、网状环等变形的网格网。

更复杂的拓扑结构包括**超立方体网**、 k 元 n 立方体网及其变形结构。一个 n 维超立方体网由 $N = 2^n$ 个结点组成,其直径等于维数,结点之间的距离小于网格网与环形网,但这种结构的高维网扩展性较差,每次扩展要增加 1 倍结点。 k 元 n 立方体网与超立方体网的区别是它的每一维包含 k 个结点,而超立方体网的每一维只有两个结点。**立方体连接环 (CCC)** 是将 k 维超立方体的每一个结点用一个含 k 个结点的环代替而形成的网,它的主要优点是每一结点的度都是 3,使网络易于扩展。高维网络每一维有专用的通信链路,不与其他维共享,因而通道的利用率低于低维网络。业已证明,对于给定对分带宽的互联网,低维网络比高维网络延迟低、冲突少、吞吐量大。90 年代推出的大规模并行机大多采用低维互联网。图 1 给出了各种静态互联网的拓扑结构,表 1 比较了各种静态互联网的特征。

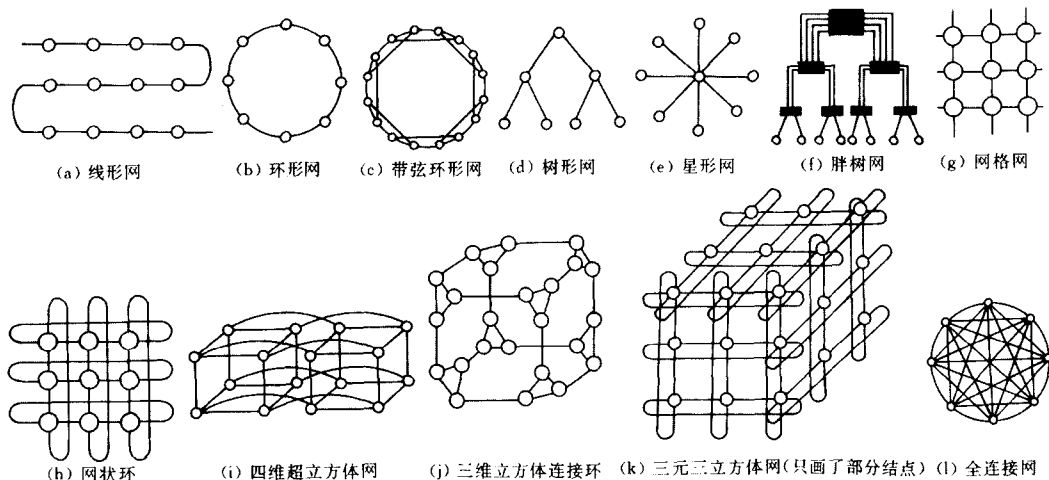


图1 静态互联网拓扑结构

表1 静态互联网的特征

网络类型	结点数	网络直径	链接数	对分带宽	是否对称	备 注
线形网	2	$N-1$	$N-1$	1	否	N 为结点数
环形网	2	$\lfloor N/2 \rfloor$	N	2	是	N 为结点数
二叉树网	3	$2(h-1)$	$N-1$	1	否	$h = \lceil \log_2 N \rceil$
二维网格网	4	$2(r-1)$	$2N-2r$	r	否	$r = \sqrt{N}$
二维网状环	4	$2\lfloor r/2 \rfloor$	$2N$	$2r$	是	$r = \sqrt{N}$
超立方体	n	n	$nN/2$	$N/2$	是	$n = \log_2 N$
CCC	3	$2k-1 + \lfloor k/2 \rfloor$	$3N/2$	$N/(2k)$	是	$N = k \times 2^k \quad k \geq 3$
k 元 n 立方体网	$2n$	$n\lfloor k/2 \rfloor$	nN	$2k^{n-1}$	是	$N = k^n$ 结点
全连接网	$N-1$	1	$N(N-1)/2$	$(N/2)^2$	是	N 为结点数

动态互联网 动态互联网包括总线、多级互联网和交叉开关网络,已运用于单指令[流]多数据流和多指令[流]多数据流并行计算机。

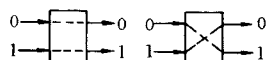
总线由一组连线及控制器件组成,供处理机、存储器模块以及外部设备传送数据之用。总线基于分时方式工作,每一时刻只允许与它相连的一对设备进行通信,当有多个通信要求同时出现时,总线仲裁器要按优先级进行仲裁。设计总线的关键技术包括总线仲裁、中断控制、一致性协议和传输进程。根据系统中各部件传输数据的不同速度,总线往往分成几个层次,直接与多处理机和存储器相连的总线速度最快,一般称为局部总线或内部总线;与外部设备相连的总线称为外部总线或系统总线(参见系统总

线)。为了便于多个功能部件通过印制板互连,多处理机的系统总线往往在计算机的底板上实现。随着技术的进步,总线的通信能力不断提高,商品化多处理机的内部总线带宽已超过 1 GB/s。

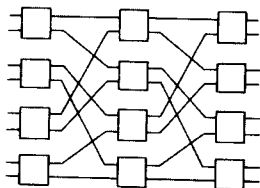
多级互联网 由多级开关模块和级间连接组成。一个 $a \times b$ 开关模块具有 a 个输入端和 b 个输出端,通常采用的开关模块规模为 $2 \times 2, 4 \times 4, 8 \times 8$ 。开关可以动态设置以建立合适的输入输出连接。开关模块允许“一对一”或“一对多”的输入输出映射,但不允许“多对一”的映射,即必须避免输出端的冲突。当只允许一对一映射时,开关模块实际上是一个 $n \times n$ 交叉开关。较常采用的级间连接方式包括全混洗、交叉开关和立方体连接等。一种较流行的多

级互联网叫 Omega 互联网,它采用 2×2 的开关模块和全混洗方式的级间连接。图 2 表示 8×8 Omega 网中 4 种可能的开关连接方式与级间互连模式。一般来讲, N 个输入的 Omega 网需要 $\log_2 N$ 级 2×2 开关模块,每一级需要 $N/2$ 个开关模块,一共需 $(N \log_2 N)/2$ 个开关模块,每个开关模块单独进行控制。IBM 公司的大规模并行计算机 SP-2 是采用多级互联网的代表。

交叉开关是连接能力最强的动态互联网。它通过一个 $n \times n$ 开关矩阵动态地实现 n 个输入到 n 个输出的任意置换。根据运行程序的要求,每一开关单元可以动态设定为开或关状态。交叉开关既可实现处理机与存储器模块之间的互连,又可以实现处理机之间的互连。例如日本富士通公司于 20 世纪 90 年代推出的 VPP 500 向量并行计算机采用 224×224 交叉开关连接处理机。处理机与存储器模块之间的交叉开关可使一个处理机同时发出对不同存储器模块的请求,实现并行存取。但对处理机之间的交叉开关,同一时刻交叉开关的每一行和每一列都



(a) 开关模块的 4 种可能的连接方式



(b) 全混洗级间连接

图 2 8×8 Omega 互联网

只允许最多 1 个开关打开。交叉开关的行类似多总线。当多个处理机同时要求存取某一存储模块或与某一处理机通信时,交叉开关必须有某种排队或仲裁机制处理冲突。

在上述 3 种动态互联网中,总线的成本最低,但每个处理机可得到的通信带宽较低,扩展性较差,因此只有规模较小的多处理机系统(一般几个到几十个处理机)采用总线互连。交叉开关是通信能力最

强但成本也最高的互联网。它的硬件复杂性与处理机数量的平方成正比。当并行计算机系统中的应用机数量很大时,采用交叉开关互连成本太高。多级互联网的通信带宽与成本都介于总线与交叉开关之间,其优点是具有较好的模块性与可扩展性,但其通信延迟与开关的级数成正比,对于 N 个处理机,则有 $\log_2 N$ 的通信延迟,而总线与交叉开关的延迟原则上都与处理机个数无关。

互联网的工作一直围绕增强网络的连接能力与降低成本两个方面进行。所谓连接能力是指网络能同时连通的最大连接数。由于竞争通信线路,有些通信请求不能同时满足,这种状态称为阻塞或冲突。上面介绍的多级互联网大都是阻塞型网络。一般,开关交叉点越多,连线越多,成本就越高,但建立连接的路径也越多,阻塞的机会越少,连接能力也就越强。交叉开关有 n^2 个开关单元,它是典型的非阻塞网络。有些互联网通过重构,即重新设置各开关模块的状态,也可以消除冲突。这种网络称为可重构非阻塞网络,但这种网络的开关控制算法相当复杂。当互联网的规模日益增大时,容错性成为另一个重要的要求,提供冗余的连接通路是实现网络容错的必要条件。

发展趋势

20 世纪 90 年代以来,采用工业上大量生产的主流处理机芯片或市场上流行的工作站构成并行计算机系统已成为发展高性能计算机的主要途径,因此处理机互连技术已成为区别不同并行计算机系统的主要标志。并行计算机系统互连水平的高低不仅体现在互联网的硬件水平上,更主要地反映在支持处理机之间的通信和协作的系统软件上,特别是路由选择、流控制、容错等有效算法的实现。互联网的设计必须考虑处理机的内部功能,尤其是互联网与处理机的接口。

有关互联网的各种工业标准已经建立或正在建立。电气和电子工程师学会(IEEE)已经通过可伸缩一致性接口(SCI)标准。这是一种基于点到点通信的高速总线标准,为研制、生产高性能的多处理机提供了一条途径。异步传送模式是下一代多媒体通信的工业标准,为语音通信和数字通信提供了高速有效的协议。未来的计算机互联网的一种方式可能会建立在异步传送模式基础上。

光通信技术发展迅速,光互连可以提供很高的通信带宽,而且噪声低、功耗小。但光开关器件的速度与集成度还赶不上电子器件,因此,在可以预见的

未来,路由选择和通道管理等开关器件仍将主要采用电子器件。电子开关器件与光通信线路的集成将是今后计算机互联网的主要发展方向。较小规模的多处理机系统主要采用电子线路互连,光互联网将首先在较大规模的并行机与远距离分布式系统中得到应用,并将逐渐成熟而成为互联网的主流。

参考文献

1. Siegel H J. Interconnection Networks for Large-Scale Parallel Processing: Theory and Case Studies. New York: McGraw-Hill, 1989
2. Hwang K. Advanced Computer Architecture. New York: McGraw-Hill, 1993 (李国杰)

huan

环 (ring) 一种具有两个二元运算的代数结构。

设 R 为非空集合且在 R 上已定义了两个二元运算加法“+”和乘法“*”。如果 R 关于“+”构成交换群,“*”对“+”满足分配律,即对任意 $a, b, c \in R$ 有 $a * (b + c) = (a * b) + (a * c)$ 和 $(a + b) * c = (a * c) + (b * c)$, 则称 R 为一个非结合环,记为 $\langle R, +, *, \cdot \rangle$ 。

在非结合环 R 中,有惟一的零元“0”且每个 $a \in R$ 都有惟一的负元“-a”,即对任意 $a \in R$ 有 $a + 0 = a$ 和 $a + (-a) = 0$ 。常把 $a * b$ 和 $a + (-b)$ ($a, b \in R$) 分别写成 ab 和 $a - b$,并约定“*”优先于“+”结合。因此,对任意 $a, b, c \in R$ 和整数 n ,恒有

$$\begin{aligned} a0 &= 0 = 0a, & a(-b) &= -ab = (-a)b, \\ (-a)(-b) &= ab & a(b \pm c) &= ab \pm ac, \\ (a \pm b)c &= ac \pm bc, & (na)b &= nab = a(nb) \end{aligned}$$

还可由归纳法证明

$$\left(\sum_{i=1}^n a_i \right) \left(\sum_{j=1}^m b_j \right) = \sum_{i=1}^n \sum_{j=1}^m a_i b_j$$

$$a_1, \dots, a_n, b_1, \dots, b_m \in R \quad \text{且 } n, m \geq 1$$

设 R 为非结合环。若对任意 $a, b, c \in R$ 有 $a^2 = 0 = (ab)c = (bc)a = (ca)b$, 则称 R 为一个李环。若对任意 $a, b \in R$ 有 $((aa)b)a = (aa)(ba)$ 和 $ab = ba$, 则称 R 为一个若尔当环。李环和若尔当环是非结合环中内容最丰富的分支。

如果非结合环 R 满足结合律,即对任意 $a, b, c \in R$ 有 $a(bc) = (ab)c$, 则称 R 为一个环。如果环 R 满足交换律,即对任意 $a, b \in R$ 有 $ab = ba$, 则称 R 为交换环。

设 R 为环。若 $a, b \in R \setminus \{0\}$ 使 $ab = 0$, 则称 a 为 R 的左零因子, b 为 R 的右零因子。它们统称为 R 的

零因子。若 $e \in R$ 使得对任意 $a \in R$ 有 $ae = a = ea$, 则称 e 为 R 的一个么元或单位元。环 R 的么元若存在,必是惟一的。如果环 R 具有么元和非零元, R 中任意非零元 a 有逆元 a^{-1} , 即 $aa^{-1} = e = a^{-1}a$, 则称 R 为一个体。交换体称为域。

全体整数、有理数和实数关于普通的加法和乘法都构成交换环,并分别称为整数环、有理数环和实数环。这些环都无零因子。

设 m 为一个正整数。模 m 的全体剩余类 $\{\overline{0}, \overline{1}, \dots, \overline{m-1}\}$ 关于模 m 的加法 $+$ 和模 m 的乘法 $*$ 构成一个交换环,称为模 m 的剩余类环,记为 $\mathbb{Z}/(m)$ 。当 m 为合数时, $\mathbb{Z}/(m)$ 有零因子。

设环 R 具有么元, $x \notin R$ 是未定元。若 $n \geq 0$ 且 $a_0, a_1, \dots, a_n \in R$, 则称 $a_0 + a_1x + \dots + a_nx^n$ 为一个 R 上 x 的多项式。全体 R 上 x 的多项式关于多项式的加法和乘法构成一个环,称为 R 上 x 的多项式环,记为 $R[x]$ 。

设 D 为环 R 的一个非空子集。若对任意 $a, b \in D$ 和 $r \in R$ 恒有 $a - b, ra, ar \in D$, 则称 D 为 R 的一个理想。对任意 $a, b \in R$ 必有

$$\begin{aligned} (a + D) \oplus (b + D) &\triangleq \{ (a + d) + (b + d') \mid d, \\ &\quad d' \in D \} = (a + b) + D \\ (a + D) \otimes (b + D) &\triangleq \{ (a + d)(b + d') \mid d, \\ &\quad d' \in D \} = ab + D \end{aligned}$$

因此 $\{a + D \mid a \in R\}$ 关于上面定义的二元运算 \oplus 和 \otimes 构成一个环,称为 R 关于 D 的商环,记为 R/D 。

设 R 和 R' 为两个环且 $f: R \rightarrow R'$ 。若对任意 $a, b \in R$ 有 $f(a + b) = f(a) + f(b)$ 和 $f(ab) = f(a)f(b)$, 则称 f 为一个环同态,并称 $\ker(f) = \{a \in R \mid f(a) \text{ 为 } R' \text{ 的零元}\}$ 为 f 的核。若环同态 f 是双射,则称 f 为一个环同构,并称 R 与 R' 同构,记为 $R \cong R'$ 。

设 R 和 R' 为两个环。若 $f: R \rightarrow R'$ 为环同构,则 $\ker(f)$ 为 R 的理想,而且当 $f(R) = R'$ 时有 $R/\ker(f) \cong R'$ (环同态定理)。

参考文献

1. Hungerford T W 著. 代数学. 冯克勤译. 长沙: 湖南教育出版社, 1985
2. 熊全淹. 近世代数. 上海: 上海科技出版社, 1978 (王兵山 王水汀)

huigui fenxi

回归分析 (regression analysis) 研究一个或多个随机变量与若干非随机变量之间关系的统计方法。考虑一个可观测的随机变量 y 与 p 个可控制的

变量 x_1, x_2, \dots, x_p 有某种函数关系,假定函数的形式未知或者函数形式已知,但其中含有某些未知的参数 $\beta_1, \beta_2, \dots, \beta_p$, 于是有 $y = f(x_1, \dots, x_p, \beta_1, \dots, \beta_p) + \varepsilon$, 其中 ε 为随机的观测误差,称 $f(x_1, \dots, x_p, \beta_1, \dots, \beta_p)$ 为 y 关于 x_1, \dots, x_p 的回归。通常假定 f 为 x_1, \dots, x_p 的线性函数,这是因为在某种情形下,比如 $y = a \sin t + b \cos u$, 只要令 $x_1 = \sin t, x_2 = \cos u$, 仍可化为线性问题来研究。另一方面对于一般光谱的函数 f , 总可以用多项式来逼近,作适当的变换也可化为线性问题。所以可设

$$y = \beta_0 + \beta_1 x_1 + \dots + \beta_{p-1} x_{p-1} + \varepsilon \quad (1)$$

写成矩阵的形式

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \quad (2)$$

其中 $\mathbf{y} = (y_1, \dots, y_n)^\top, \boldsymbol{\beta} = (\beta_0, \beta_1, \dots, \beta_{p-1})^\top, \boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n)^\top$

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & \dots & x_{1,p-1} \\ \dots & \dots & \dots & \dots \\ 1 & x_{n1} & \dots & x_{n,p-1} \end{bmatrix}$$

为 $n \times p$ 阶的已知矩阵, $E\boldsymbol{\varepsilon} = 0, \text{var}(\boldsymbol{\varepsilon}) = \sigma^2 \mathbf{I}_n$ 。这里所谓线性是对参数 $\boldsymbol{\beta}$ 而言的。

回归分析的主要问题是: ①如何估计 $\beta_0, \dots, \beta_{p-1}$ 以及 σ^2 ; ②对有关 $\boldsymbol{\beta}$ 的某种假设和模型 (2) 的假设进行检验; ③对 y 进行预测或控制。

参数的最小二乘估计 考虑误差平方和

$$Q = \boldsymbol{\varepsilon}^\top \boldsymbol{\varepsilon} = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

其中 $\hat{y}_j = \sum_{i=0}^{p-1} \beta_i x_{ji}, j = 1, 2, \dots, n$ 称为估计值。使得 Q 达到最小的估计称为最小二乘估计, 它们是

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}, \quad \hat{\sigma}^2 = \frac{1}{n} Q$$

它们都是极大似然估计, 而 $\hat{\sigma}_e^2 = \frac{1}{n-p} Q$ 为 σ^2 的无偏估计。

对 y 的预测与控制 给定置信水平 $1 - \alpha$, 令

$$\left. \begin{aligned} \hat{y}_1 &= \hat{y} - \hat{\sigma}_e t_{n-p}(\alpha) \left(1 + \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_i x_j \right)^{\frac{1}{2}} \\ \hat{y}_2 &= \hat{y} + \hat{\sigma}_e t_{n-p}(\alpha) \left(1 + \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_i x_j \right)^{\frac{1}{2}} \end{aligned} \right\} \quad (3)$$

则有 $p(\hat{y}_1 \leq y \leq \hat{y}_2) = 1 - \alpha$, 式中 $\hat{y} = (\hat{y}_1, \dots, \hat{y}_n)^\top$,

$\hat{y}_j = \sum_{i=1}^p \hat{\beta}_i x_{ji}, x_i$ 为 \mathbf{X} 中第 i 个列向量, c_{ij} 为 $(\mathbf{X}^\top \mathbf{X})^{-1}$ 中元素, $t_{n-p}(\alpha)$ 为 t 分布的临界值。由此,

如要控制 \hat{y}_1, \hat{y}_2 的值, 便可确定相应的 x_i 的值。

线性模型的假设检验 检验所提的线性模拟是否合适。为此令 $H_0: \beta_1 = \beta_2 = \dots = \beta_p = 0$ 。如果 H_0 不被否定则说明模型不合适或拟合的效果甚差。由于 H_0 成立时, 统计量 $(n-p)U/(pQ_e) \sim F(p, n-p)$ 分布 (这里 $U = \mathbf{y}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}, Q_e = \mathbf{y}^\top (\mathbf{I} - \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top) \mathbf{y}$), 于是得到拒绝域为 $(n-p)U/pQ_e > F_{p, n-p}(\alpha)$ 这里 $F_{p, n-p}(\alpha)$ 是 $F_{p, n-p}$ 分布的临界值。

为检验某个自变量 (如第 i 个) 对 y 的影响甚微, 可提出 $H_0: \beta_i = 0$, 在 H_0 成立下, $F \triangleq (n-p)\beta_i^2/(Q_e c_{ii}) \sim F_{1, n-p}(\alpha)$ 。由此可确定拒绝域为 $F > F_{1, n-p}(\alpha)$ 。

举一个例子, 有人从钢线的碳含量 x (%) 与 20℃ 时的电阻值 y ($\mu\Omega$) 的测量数据中, 得到回归方程

$$\hat{y} = 13.9584 + 12.5503x$$

于是便可预测, 当碳含量为 50% 时, 钢线的阻值以 95% 的置信度处于 19.66 $\mu\Omega$ 到 20.81 $\mu\Omega$ 之间。

参考文献

陈希孺. 数理统计引论. 北京: 科学出版社, 1981
(金治明)

huibian chengxu

汇编程序 (assembler) 把汇编语言书写的源程序翻译成等价的机器语言程序的处理系统。

汇编程序以汇编语言书写的源程序作为输入, 以机器语言表示的目标程序作为输出, 其主要工作过程是: ①输入汇编语言源程序; ②检查语法的正确性, 如果正确, 则将源程序翻译成等价的二进制或浮动二进制机器语言程序, 并根据用户的需要输出源程序与目标程序的对照清单; 如果语法有错, 则输出错误信息, 指明错误的部位、类型和编号; ③对翻译出的目标程序进行必要的善后处理, 如直接启动执行, 以可执行程序段形式保存在外存中, 与外存中其他程序段或程序库中的程序连接装配成完整的目标程序。

在广泛使用汇编程序的过程中, 人们不断地吸收其他处理系统的优点, 相继研制出模块汇编程序、宏汇编程序、高级汇编程序、条件汇编程序等各具特色的汇编程序。模块汇编程序吸收了模块程序设计思想, 提供模块单独汇编功能, 支持模块的并行设计、编程、调试和连接装配等能力。宏汇编程序的特点是在汇编程序的基础上增加宏加工程序功能, 允

许用户在源程序中方便地定义和使用宏指令。高级汇编程序的基本思想是用汇编语言编写源程序中的数据加工部分,而用高级语言的控制语句编写控制部分。这样既保持了汇编语言的优点,又吸收了高级语言易于书写和阅读的长处。条件汇编程序允许用户在源程序中设置条件汇编指示,并通过预置不同参数值的方法灵活地剪裁(选择或跳过)、汇编不同的程序段。

鉴于汇编语言的指令与机器语言的指令大体上保持一一对应的关系,所以汇编程序通常采用两遍扫描源程序的实现算法。第一遍查明源程序中符号的定义和使用情况,并将有关信息收集到符号表中;第二遍利用符号表中的信息,将源程序中的符号化指令逐条翻译成相应的机器指令。如果汇编语言中规定“所有符号一定要先定义,后使用”,则可以容易地将两遍算法合并成一遍算法实现相应的汇编程序。汇编程序的具体翻译工作可归纳为如下几项:①用机器操作码代替符号化的操作符;②用数值地址代替符号名字;③将常数翻译为机器的内部表示;④分配指令和数据所需的存储单元。除上述翻译工作外,汇编程序还应考虑如下工作:①处理汇编伪指令,收集程序中提供的汇编指示信息,并执行相应的功能;②向用户提供汇编过程中的有关信息以及源程序与目标程序的对照清单;③根据用户要求和汇编后程序段的特点,执行不同的善后处理工作,如直接执行、记入外存保留、连接装配等;④如果汇编语言支持模块汇编、宏汇编、高级汇编或条件汇编等功能时,汇编程序则应提供相应的处理任务。

(曹东启)

huibian yuyan

汇编语言 (assembly language) 将机器语言中地址部分符号化并可进一步包括宏构造的语言。

汇编语言由汇编指令和汇编伪指令两部分组成。汇编指令是机器指令的地址部分符号化表示,其操作码采用易记的操作符表示,而地址码则采用标号、变量名字、常数等直观的表现形式。汇编指令基本上与机器指令保持一一对应的关系。在汇编过程中,它将被翻译成对应的机器指令;运行时,它将执行相应机器指令所规定的功能。汇编伪指令又称作汇编指示,其作用是指示汇编程序如何进行汇编,用于向汇编程序提供用户自定义的符号、数据类型、数据空间长度、目标程序格式、数据或指令的存放位

置等提示信息。采用汇编语言编写程序虽不如高级语言简单、直观,而且必须对机器内部结构和特性有较多的了解;但它占用内存少,运行效率高,且能直接控制各种设备资源。因此,汇编语言经常用于编写大型软件系统的核心部分程序,或者用于编写运行时间长或实时性要求高的程序部分。

在不断吸收先进编程思想和相关软件优点的基础上,发展、演变出各具特色的汇编语言。模块汇编语言是在模块程序设计思想指导下,以模块作为编程基本单位而设计的汇编语言,它支持单个模块独立汇编和多个模块联合汇编的功能。宏汇编语言是在汇编语言的基础上,增加宏定义和宏调用功能而形成的汇编语言,它将为用户提供自定义指令的功能。高级汇编语言是在汇编语言的基础上,增加高级语言中控制语句成分(如条件语句、循环语句、函数和过程等)而成的汇编语言,它既保持汇编语言的有效性和灵活性,又充分发挥了高级语言简单、直观、易于编写等优点。条件汇编语言是在汇编语言中引进“条件转移”和“无条件转移”等汇编指示。这将为用户提供一种简便、灵活的剪裁(选择或跳过源程序)的手段。

(曹东启)

huihe

会合 (rendezvous) 两任务间的一种同步机制。又称汇合。**Ada 语言**中两个任务的相会。为了使两个任务同步和通信,两个任务必须相会,先到达的要等后到达的,两者都到达了,会合就开始。**Ada**的会合双方是不对称的:一个主动任务,一个被动任务。主动任务通过入口调用发出约请,而被动任务通过接受语句表示接受约请,准备会合。在任务规约中有入口声明任务,从这个入口的意义上说,是被动任务,因为这个入口是为另外的任务准备的,由另外的任务发出对这个入口的调用(一个任务如果调用自己的入口,必然会产生死锁)。会合后,两任务独立地继续它们各自的运行。

会合概念最早是由 J. Conway 在 1963 年提出的,它结合了同步与消息传递。这种机制后来由 C. A. R. Hoare 与 B. Hansen 重新肯定,这是第一种高级同步机制,在 **Ada 语言**中已经采用,对于分布式系统来说,这种同步机制比较合适。

参考文献

徐家福. 系统程序设计语言. 北京: 科学出版社, 1983

(程虎)

huituji

绘图机 (plotter) 一种能在纸张、薄膜和胶片等记录介质上绘出计算机生成的各种图形或图像的设备。绘图机最早出现于 20 世纪 50 年代末,最近十年发展很快,每年都有新型号的绘图机推向市场。目前各种性能规格的绘图机已广泛应用在各行各业和各种科学技术领域之中,成为图形、图像处理系统不可缺少的输出设备。

绘图机的分类

绘图机的类型很多,可以按外形结构、驱动方法、控制系统和介质上图形绘制的方法等进行分类。根据介质上的图形绘制方法的不同可将绘图机分为向量绘图机和点阵绘图机两大类。

向量绘图机以线段作为构成图形的基本元素,这些线段是用绘图工具,也就是包括圆珠笔、墨水笔、铅笔等在内的各种笔,以及刻刀、刻针、光笔等在媒体上画出来的,所以又将向量绘图机称为笔式绘图机。向量绘图机在结构上有滚筒和平板两种类型。点阵绘图机则以点作为构成图形的基本元素,实际是用点阵图像表示图形,由于图形不是用笔绘制出来的,所以也称为无笔绘图机。这两种绘图机都能按计算机的命令绘出各种图形。但是向量绘图机不能接收和处理计算机输出的点阵绘图命令,而

点阵绘图机不管计算机输出的是点阵绘图命令还是向量绘图命令,它都用点阵表示出来。

绘图机的工作原理

滚筒式向量绘图机的工作原理 该绘图机的基本结构如图 1 所示。其工作原理如下:将绘图介质用具有很大摩擦力的压轮压在旋转的滚筒上,驱动器驱动一只电机带动这个滚筒滚动时,也就带动绘图媒体在它的 x 轴方向运动,另一只电机带动笔架在 y 轴方向运动;同时由笔驱动电路驱动笔架中的笔抬落机构在 z 轴方向运动,在需要绘图的地方落笔,在不需要绘图的地方抬笔,画出想绘的图形。在绘制不同粗细或不同颜色的线条时,绘图机能自动地到笔库中换笔。一般在笔库中可安装 4 支到 16 支不同粗细或不同颜色的绘图笔供自动选用,所以绘图机能在了一幅图内绘出多种宽度和多种颜色的线条。滚筒绘图机中有些型号对介质的长度没有限制,如果采用成卷的介质和相应的输纸机构,在连续绘图时可以不逐张更换介质。滚筒绘图机的结构简单,速度和加速度高,占地面积小,使用方便,价格也比较低廉,但绘图精度相对低一些。适用于对图形精度要求不太高,但出图速度要求较高的场合。近几年来滚筒式向量绘图机的发展很快,也有不少高精度产品问世。

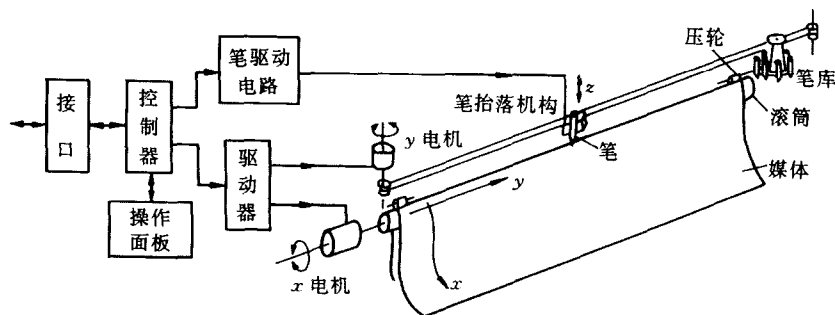


图 1 滚筒式向量绘图机结构示意图

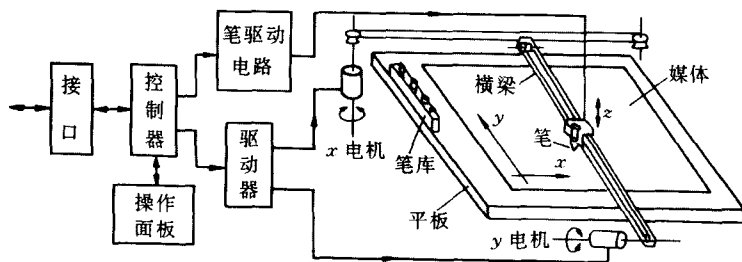


图 2 平板式向量绘图机结构示意图

平板式向量绘图机的工作原理

该绘图机的基本结构如图 2 所示,其工作原理如下:采用静电吸附或磁性压条等方法将绘图媒体固定在平板上,一只电机推动横梁在 x 轴方向运动,另一只电机推动横梁上的笔架在 y 方向运动,其他绘图笔在 z 轴方向的抬落、换笔等与滚筒式向量绘图机相同。平板式向

量绘图机的机械结构较为复杂,由于横梁的重量较重,对控制系统的要求也就较高,速度、加速度也相对较低。平板式向量绘图机可用许多方法将精度做得很高,但价格也随精度要求提高而上升。高精度平板式向量绘图机适用于对图形精度要求较高的场合。

点阵绘图机的工作原理 该绘图机的基本结构如图3所示,由图可见,点阵绘图机结构是滚筒式的,它与滚筒式向量绘图机相比只是将向量绘图机的笔

抬落驱动器换成了点阵写入电路,笔架改为点阵写入头。如果点阵绘图机从计算机接收到的是点阵数据,则由控制器直接将这些数据送往点阵写入电路,由写入头写入介质;如果从计算机接收到的是向量绘图命令,就要经过向量点阵变换器将向量变换成点阵数据送入点阵写入电路。向量点阵变换器可以用控制器中的程序实现,但为了提高变换速度,一般采用硬件实现。

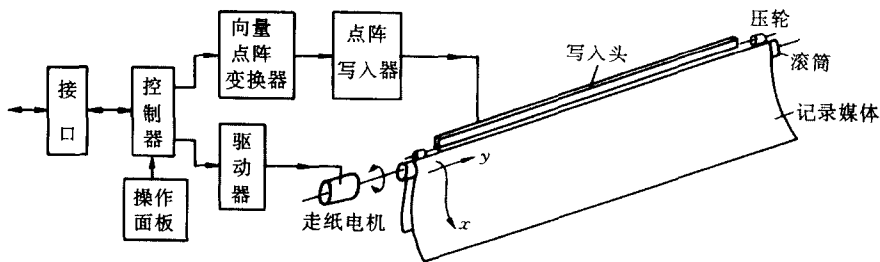


图3 点阵绘图机结构示意图

点阵绘图机的主要优点是不管图形的复杂程度如何,出图的速度几乎一样,在图形较为复杂时,出图的速度要比向量绘图机快得多。点阵绘图机绘出的图形画面色彩丰富,适合表现三维立体图形的表面涂色、光照等效果以及计算机的图像输出。点阵绘图机的工作噪声很小。

利用静电、喷墨、激光、热转印和发光二极管等各种在介质上写入点阵的技术都可以制成点阵绘图机。

绘图机控制器和接口

不同类型的绘图机中的控制器和接口的基本功能都一样。

绘图机的控制器是以微处理器为核心构成的控制系统,在它上面运行一个固化了的控制程序。控制器一方面要接收用户从操作面板键入的命令,另一方面逐条解释从计算机送来的绘图命令,根据这些命令的要求控制绘图机工作,完成绘图任务。从计算机送来的绘图命令中通常包括直接绘图命令和控制命令,前者如绘制线段、曲线、文字、符号和点阵数据等,后者如换笔,改变速度和加速度,改变笔压力等。这些命令的集合就是绘图机的绘图语言(也称为绘图机的命令集)。目前为了解决兼容问题,绘图语言已逐步标准化。

为了能和不同的计算机相连接,绘图机一般都选用某种标准的设备接口,如RS-232串行接口。

绘图机的主要性能指标 向量绘图机的主要性

能参数定义如下:

(1) 分辨率 绘图工具沿 x 轴或 y 轴运动的最小长度值,或称为绘图机的步距。

(2) 有效绘图幅面 确保绘图工具正确绘图的最大区域,用长 \times 宽表示。

(3) 绘图速度和加速度 绘图工具相对绘图介质沿轴向运动的最大速度和加速度。

(4) 绘图精度 由机械系统、控制系统和换笔等因素引起绘制图形的实际尺寸值与理想尺寸值之差,也称定位误差。

(5) 笔数 笔库中可同时安放的笔的支数。

中华人民共和国国家标准GB 9814—88《向量绘图机通用技术条件》中对向量绘图机的等级和档次作了规定。

一般绘图机的速度、精度和有效绘图幅面是相互制约的性能指标。例如,绘图精度要求很高时,有效绘图幅面就不能做得很大,同时速度、加速度也不能提得很高。

点阵绘图机的性能指标中的分辨率、精度、速度和色彩等的定义和规定都与相应的点阵印刷设备(如针式打印机)基本相同。点阵绘图机的有效绘图幅面、接口和命令集等与向量绘图机的要求和规定基本一样。

向量(笔式)绘图机,特别是光绘图机、刻图机等专用笔式绘图机将会有较快的发展;点阵绘图机由于突破了关键技术,有可能后来居上。绘图机技

术发展的趋势是提高绘图机的图形处理能力和绘制能力;增加数据吞吐量和设备的通用性;提高绘图机的精度和色彩表现能力,使图形更加精确,色彩更加丰富;提高出图速度,从而提高工作效率;绘图命令集进一步扩展,并逐步标准化,从而改善软件的兼容性;扩大随机存取存储器(RAM)缓冲或增加磁盘缓冲存储,增强脱机重绘的功能;扩大自诊断功能和人机交互操作能力。此外,笔式绘图机还要进一步降低各种因素引起的噪声。

参考文献

1. 郭平欣,姚锡珊,虞浦帆主编. 电子计算机外部设备原理. 北京: 国防工业出版社, 1986
2. 张江陵,季国钧等. 电子计算机外部设备设计原理. 武汉: 华中理工大学出版社, 1989

(周利华)

hunhe guangxian tongzhou dianlan

混合光纤同轴电缆(hybrid fiber coaxial cable, HFC) 基于有线电视(CATV)网的光纤光缆和同轴电缆混合网的宽带接入体系结构。它是以同轴电缆网络为最终接入部分的宽带网络系统。

HFC的网络结构是在头端到光节点的光网络单元之间通过光纤光缆传输光波信号,光节点到用户家之间则利用原有的同轴电缆分配网来传输和分配用户信息。它既保留了传统的模拟传输方式,又实现了上、下行双向和模拟、数字混合传输,能同时提供电话、模拟视频、数字视频和交互式业务。

HFC宽带接入网涉及的关键技术:

(1) 上行通道噪声与干扰的抑制技术 由于HFC的同轴电缆部分是一点到多点的树形结构,反向通道的噪声是各支路放大器的级联噪声和各支路间噪声的叠加,形成了噪声的漏斗效应。反向通道噪声源有四类,即窄带短波噪声,频带为5~30 MHz;冲击噪声,频带为60~2 MHz;由设备的非线性引起的共模失真;以及用户在使用各种电器时产生的频带在30 MHz以下的本地干扰噪声。抑制噪声的措施有以下几点:选用屏蔽特性好的电缆,以消除从空间耦合进入电缆的噪声干扰;选用连接特性好,无泄漏电磁波的电缆连接器,以消除因连接泄漏而引入的噪声;室内电缆端口用匹配器连接,防止从端口接收和发射干扰;及时更换接触不良或锈蚀的电缆及接头;选择经严格培训合格的人员精心操作安装。

(2) 上行通道多址接入技术 在HFC上行通

道中,常用三种多址接入技术,即时分多址接入(TDMA)、频分多址接入(FDMA)和码分多址接入(CDMA)。时分多址是将用户的数据流分配到特定的时隙;频分多址是每个用户的数据流都有一个载频;码分多址是为每个用户分配一个特定的扩展序列。每一种多址接入技术对CATV上行通道的干扰承受能力是不同的。对于多址接入的干扰(MAI),TDMA和FDMA由于信号排列在时隙或频率隙内,相邻时隙或频率隙影响可以忽略不计,而CDMA的容量则受MAI的限制。

(3) 下行通道的数模混合传输技术 HFC的CATV网,下行不仅传送多路模拟视频信号,还传送多路数字电视,因此,下行信息是一个模拟、数字混合传输的信道。设计时,既要保证模拟信道的指标要求,又要满足数字信道误码率要求。模拟信号和数字信号的叠加,会产生尖峰脉冲。数字信号的引入要保证不影响模拟视频信号,总的光调制指数保持不变。模拟视频信号的限幅失真会使数字信号的误码率大大增加,采用合适的编码方式、调制方式和纠错技术,可以减少限幅噪声对数字信号的干扰,保证数字信号的误码率达到要求。

HFC可以充分利用现有的CATV网宽频的特点,不需重新敷设配线网。除了提供有线电视节目外,还可为用户提供电信业务和宽带交互式多媒体业务。HFC尚需解决的问题有标准化,网络的同步、网管、信令,噪声的抑制,安全保密等。

参考文献

1. 胡道元. 智能建筑计算机网络工程. 北京: 清华大学出版社, 2002
2. Padmanand warrier, Balaji Kumar, XDSL Architecture. McGraw-Hill Inc., 2000 (胡道元)

hunhe jisuanji

混合计算机(hybrid computer) 把模拟计算机与数字计算机结合在一起,应用于系统仿真的计算机系统。它兼有模拟计算机解题速度快和数字计算机计算精度高、逻辑和存储功能强的优点。早期的混合计算机是将通用的、但具有一定逻辑和存储功能的模拟计算机通过混合接口(或称中间界面)与通用数字计算机组合而成,故也称“组合式”混合计算机系统。这种系统即使采用良好的转换装置,效率仍然不高,软件开发也很困难。20世纪70年代以后,研制开发了新型的混合计算机,从整体考虑来设计系统的软、硬件,使其性能和效率都大为

提高。

混合计算机一般由 3 部分组成: 通用模拟计算机, 数字计算机和混合接口(中间界面), 如图 1 所示。图中的中间界面与通用模拟计算机一起, 有时也统称为混合模拟计算机。模拟计算机在混合计算机系统中主要承担快速运算任务。数字计算机则承担对系统的监督控制和数据处理, 产生模拟计算机

所需的迭代运算程序, 对某些复杂函数(如多变量函数, 坐标变换, 高精度积分等)进行高精度运算等任务。中间界面(混合接口)的作用除了将模拟计算机和数字计算机的变量通过模数转换器和数模转换器进行转换之外, 还有逻辑功能与控制信号的传递, 以完成并行运算的模拟计算机与顺序运算的数字计算机在时间上的同步。

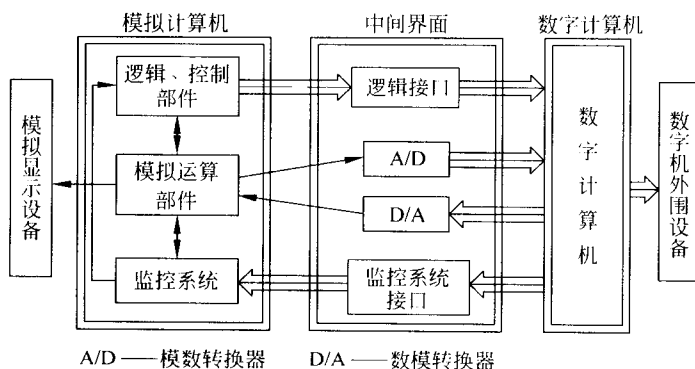


图 1 混合计算机的组成

随着微电子技术的迅速发展, 混合计算机已发展成为具有自动编排仿真程序的多处理机系统。这种系统具有更强的实时仿真能力。混合计算机主要应用于严格要求实时的复杂大系统的仿真, 尤其在航空航天方面的应用, 例如导弹系统、航天飞行器系统等的仿真。

参考文献

孙国基, 李人厚. 计算机仿真技术. 北京: 国防工业出版社, 1980

(李人厚)

hunhe jisuan moxing

混合计算模型 (hybrid computational models) 计算机与其所控制的物理部件构成的, 具有既随时间连续变化的变量又受事件驱动的离散变量的系统的数学模型。

混合系统的设计涉及控制理论和计算机科学, 20 世纪 90 年代以来引起了计算机科学界很大关注。很多混合系统要求绝对安全, 如自动导航系统, 核电站监控系统等。绝对安全系统的设计是计算机软件科学的重大课题。当前的软件产品耗资巨大, 但多数无法避免差错。这样的软件不能用于绝对安全系统。计算机软件科学界提出使用严格的形式化方法来设计该类软件。

迄今, 形式化方法所处理的对象都是离散变量(或已离散化的变量), 所用的语言及演算亦都是基于离散数学的公理化系统。而混合系统设计的形式化方法不可回避连续数学; 控制理论使用的是微分方程刻画连续变量的变化规律。混合系统设计的形式化方法也不能回避离散的事件, 这些事件驱动系统的变化。混合系统设计的形式化方法需要一种计算模型, 它同时支持连续变量和离散事件耦合系统的计算。

近几年在已有计算理论的基础上, 已陆续发展了多种混合计算模型。大体上可分为逻辑型、程序设计型和自动机型三类。

逻辑型混合计算模型的主要思想基于时态逻辑, 引入时段和切变的概念。时段可用来刻画系统在一个时间区间上的连续变化, 而切变则表示事件的发生(离散变量的变化)。在单个时段上, 借用连续数学(微分方程理论)推导系统的行为; 而在相邻时段间, 则用时态逻辑中切变算子的规则, 推导系统行为的转化。逻辑型计算模型中的时段演算, 已引起该领域同行的广泛重视。该演算是由周巢尘, C. A. R. Hoare 和 A. P. Ravn 所建立。

程序设计型混合计算模型是将传统的程序设计语言加以推广以容纳连续变量。推广后的程序语言可用来描述混合系统的行为。而其中的控制

部分可逐步求精,变换成传统的可在计算机上执行的软件,从而生成数值控制系统。通信顺序进程 CSP,已推广为混合通信顺序进程。在这个程序语言中,有一种特殊的语句称为连续构件,它可表示一个具体给定初值的微分方程;而原有的通信语句可用来表达事件的起源和发生;程序语言中的顺序算子,条件算子等用来刻画连续构件和通信间的耦合关系。

自动机早已用于各种模拟计算系统,计算机本身亦可看作一个庞大的有限状态自动机:一个状态表示计算机中各存储器和寄存器一种取值,而计算机的操作导致计算机由一个状态转移至另一个状态。如果将自动机的状态看作是在一组微分方程控制下,一组连续变量的连续变化过程,则将状态的转移视作事件的驱动。这种推广后的自动机称作混合自动机,可用来描述和计算混合系统的行为。

总之,混合计算模型还在发展完善之中。

参考文献

1. Zhou Chaochen. A Calculus of Durations. Information Processing Letters, 1991, 40(5)
2. He Jifeng. From CSP to Hybrid Systems, A Classical Mind. Prentice-Hall, 1994
3. Alur R. Hybrid Automata. Hybrid Systems, LNCS 736, Springer-Verlag, 1993 (周巢尘 李晓山)

hunhe zidongji

混合自动机 (hybrid automaton) 一种描述混合系统的计算模型。它是有限状态自动机的推广,可用来描述和计算具有连续和离散变量的混合系统的行为。在混合自动机中,状态被看作是在一组微分方程控制下,一组连续变量的连续变化过程;而将状态的转移视作事件的驱动。

混合自动机最早是由美国 R. Alur, T. A. Hensinger 和 Z. Maznna 等在 20 世纪 90 年代以后提出,并给出了一些有关理论结果,如可达性问题的不可判定性,以及受限的线性混合自动机的一些性质。下面我们用例子简要介绍一下混合自动机。

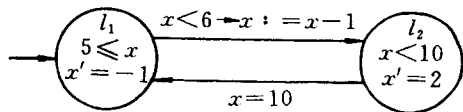


图 1

图 1 表示一个线性混合自动机,且只有一个数

据变量 x 。它所描述的系统拥有两个状态位置 l_1 和 l_2 。在位置 l_1 中, x 的数值减少的速度为 1, 即 $x' = -1$ 。从位置 l_1 到位置 l_2 的迁移, 在 $5 \leq x \leq 6$ 的任何时刻都可能发生。当迁移发生时, x 的数值即刻减 1, 进入位置 l_2 。一旦进入位置 l_2 , x 将以 2 的速度增加, 即 $x' = 2$ 。只要 x 的数值一达到 10, 迁移发生, 进入位置 l_1 。事实上因为位置 l_2 具有不变式 $x < 10$, $x = 10$ 时, 系统已经在位置 l_1 了。

用混合自动机描述混合系统的优点在于直观, 其缺点是不易进行推导和求精。

参考文献

- Alur R. Hybrid Automata. Hybrid Systems, LNCS 736, Springer-Verlag, 1993 (周巢尘 李晓山)

huoqu guocheng

获取过程 (acquisition process) 获取者获得一个系统或软件产品的一系列活动。它从确定获取该系统或软件产品的需求开始, 经过招标准备、合同准备、谈判及修改、对供应方的监督等活动, 直至验收完成方告结束。获得该系统或软件产品的机构可就部分或全部获取活动与某机构签订合同, 后者根据获取过程开展相应的活动。二者皆可称作获取者。

获取者可根据项目具体情况对这些活动进行剪裁和组织(参见剪裁过程)。

获取过程主要包括以下各项活动:

(1) 开始 获取者确定要取得的一个系统或软件产品, 并对其需求进行定义, 选择获取方式, 制定计划、验收策略、准则和条件。它有以下具体任务: ①系统需求定义。按照开发过程中有关活动内容来进行。需求定义中最好要包括安全性、保密性和其他关键需求以及要求设计、测试需遵循的有关标准和规程。如果由供应者完成需求定义, 则结果必须要经获取者的同意和认可。②获取方式的选择。获取者应根据项目的具体任务和风险大小来选取获取方式, 获取方式一般有以下几种: (a) 购买现货产品; (b) 自行开发; (c) 通过合同委托开发; (d) 上述三种方式的结合; (e) 改进现有的软件产品。当选用方式(a)时, 该现货产品必须要具备能满足需求、有必备的文档、能达到所有权和使用权要求, 并有未来的产品支持计划等条件。如有多个现货产品可供选用时, 最好还要按有关评价选择标准等作为准则, 进行优选。③以文档形式制定获取计划。计划中要包括系统需求、使用定义、执行合同的类型、

所涉及机构的责任、将使用的支持概念、预计风险以及解决这些风险的办法等内容。④确定验收策略和准则。

(2) 招标准备 ①根据开始活动中选用的获取方式编写一份系统获取需求的文档,即标书,其中要包括系统需求、项目描述、投标者须知、软件产品清单、子合同或合同条款、技术制约(如目标环境)等内容。②根据项目决定采用哪些过程和活动,并对它们进行相应剪裁,还要特别指明可应用的支持过程和它们的执行机构,以让供应者在投标书中确定各个特定支持过程的执行方法。③确定供应者在供应过程中将接受评审和审计的合同阶段目标,并作为对他们进行监督的一部分(参见支持过程)。④把获取需求文档交给实施获取活动的有关机构。

(3) 合同的准备和修改 ①确定选择供应者的方法和步骤,其中要包括对投标书的评价准则和符合需求的程度。②对供应者的投标书、能力及其他需要考虑的因素进行综合评估,并以此为基础选择一个合适的供应者。③按项目剪裁各项活动及内容,并将最终决定包含于合同内容中。④准备一份合同,并就此与供应方商议和谈判,商议时要提出系统的需求、成本和交付软件产品的日程,合同中还要涉及可供使用的现货产品的产权、使用权和所有权等条款。⑤合同一旦生效,获取者将通过与供应者谈判来控制合同的修改,同时要了解该修改对项目计划、成本、质量和日程等的影响。

(4) 监督供应者 获取者将进行支持过程中的联合评审和审计过程,对供应者实施监督,必要时还要采纳支持过程中的验证和确认过程(参见支持过程)。在整个获取过程中,获取者要与供应者密切合作以便及时提供全部必要的信息和解决所有有待共同解决的问题。

(5) 验收完成 根据原先已确定的验收策略、准则和条件为验收作好全部必要的准备,诸如测试用例、测试数据、测试规程和环境准备等等,然后对交付的软件产品进行验收评审和测试,当产品符合所有的验收要求后,便予以接受。验收后,需要时还可对已交付的软件产品进行配置管理(参见支持过程)。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes —IEEE Std 1074 ~ 1991
2. ISO /IEC 12207:1995 Information Technology

—Software —Part 1: Software Life -Cycle Process

(陈森芬)

Huoen luoji

霍恩逻辑 (Horn logic) 一阶逻辑的子部分。原子公式称为正文字,原子公式的否定称为负文字,它们统称为文字。例如, $P(a)$, $P(f(x))$ 是正文字, $\neg Q(a, f(x))$ 是负文字。形式为 $L_1 \vee \dots \vee L_m$ 的公式称为子句,其中 L_1, \dots, L_m 是文字。用 \square 表示不包含任何文字的空子句。它是一个永假式,即总是解释为假命题的公式。至多包含一个正文字的子句称为霍恩子句,是逻辑学家 A. Horn 于 1951 年首先研究的。例如, $P(a)$, $\neg P(a) \vee \neg Q(a, f(x))$, $P(a) \vee \neg P(x)$ 都是霍恩子句,而 $P(a) \vee P(f(a))$ 却不是霍恩子句。

霍恩逻辑是由霍恩子句组成的一阶逻辑的子部分。它是逻辑程序设计的理论基础。在逻辑程序设计中采用了一种特殊的子句记号,用 $A_1, \dots, A_k \leftarrow B_1, \dots, B_n$ 表示子句 $A_1 \vee \dots \vee A_k \vee \neg B_1 \vee \dots \vee \neg B_n$ 。霍恩子句共分以下四种形式: ① $A \leftarrow B_1, \dots, B_n$, ② $A \leftarrow$, ③ $\leftarrow B_1, \dots, B_n$, ④ 空子句 \square 。其中①称为过程,②称为事实,③称为目标。

霍恩子句可以表达知识库和对知识库的询问。例如,已知“张三在哪儿,他的狗就在哪儿”,和“张三在火车上”,询问:“张三的狗是否也在火车上?”。用个体常元 Zhang, train, dog 分别表示张三、火车、张三的狗,用 $AT(x, y)$ 表示 x 在 y 上,则“张三在哪儿,他的狗就在哪儿”表示为过程 $AT(dog, x) \leftarrow AT(Zhang, x)$ “张三在火车上?”表示为事实 $AT(Zhang, train) \leftarrow$, 问题“张三的狗是否也在火车上?”表示为目标 $\leftarrow AT(dog, train)$ 。因为上述三个子句的集合没有模型(即没有使它们都有效的结构),因此问题的答案是肯定的,即张三的狗确实也在火车上。

Prolog 语言是以霍恩逻辑为基础的高级程序设计语言。一个 Prolog 程序实际上就是一个由事实和过程组成的霍恩子句集,询问就是一个目标。运行一个 Prolog 程序就是通过计算机判断由程序和目标组成的霍恩子句集是否有模型。

事实上,要判断一个霍恩子句集是否有模型,不必考虑所有的结构,只需考虑一类特殊的结构,即埃尔布朗结构就够了。有模型的霍恩子句集必有埃尔布朗模型。设 S 是一个霍恩子句集,用 S 中出现的个体常元和函数符号构造出来的所有无变元项称为 S 的埃尔布朗域(如果 S 中不出现个体常元,就增加

一个个体常元 a)。例如, 设 S 由以下霍恩子句组成:

$$P(x) \leftarrow Q(f(x), g(x))$$

$$R(y) \leftarrow$$

则 S 的埃尔布朗域为 $\{a, f(a), g(a), f(f(a)), f(g(a)), \dots\}$ 。满足以下两条件的结构称为埃尔布朗结构: ①论域是埃尔布朗域; ②每个无变元项解释为自己。

如果两个公式集有相同的模型, 则称它们是等价的。一个公式集等价于一个霍恩子句集的充分必要条件是: ①该公式集的模型的子结构仍是它的模型; ②该公式集的模型的直积仍是它的模型。

由前提 A_1, \dots, A_n 能推出结论 B 当且仅当公式 $(A_1 \wedge \dots \wedge A_n) \rightarrow B$ 是逻辑有效式。因此, 研究判定公式的逻辑有效性的算法在人工智能中占有十分重要

的地位。一闭公式 A 是逻辑有效式当且仅当 $\neg A$ 是不可满足式, 即没有模型的公式。对于每个闭公式 A , 可找到一个有穷子句集 S , 使得 A 不可满足当且仅当 S 不可满足。因此, 判定公式的逻辑有效性的问题可以归约为判定有穷子句集的不可满足性问题。用归结法证明有穷子句集不可满足的过程可以表述为一组霍恩子句的反驳。因此, 对于任何有穷子句集 S , 可找到一组霍恩子句集 S' , 使得 S 不可满足当且仅当 S' 不可满足。

参考文献

1. Kowalski R. Logic for Problem Solving. New York: North Holland, 1979
2. Lloyd J W. Foundations of Logic Programming. Berlin: Springer-Verlag, 1984 (何自强)

J

jidashi dayinji

击打式打印机 (impact printer) 利用机械击打动作使字模隔着色带在纸上印出字来的设备。它是最早研制成功的计算机印字设备。

击打式印字设备分为整字形击打、点阵击打、点阵串行击打及点阵并行击打等 4 类。

整字形打印设备基于机电原理,利用电磁铁驱动字锤,隔着纸和色带向选定的字模上击打印字。不同字符的完整字形的字模每击打一次印出一完整字形,字模安置在各种形态的载体上,靠字模载体的机械运动来变换处于字锤击打位置的字符。有的字模本身就起着字锤的作用,隔着色带与纸向平台或圆柱上击打印字。整字形击打印字的优点是印字美观自然。其缺点是噪声大,印字速率低,字符种类少,字体更换不便或不可能,机械磨损问题突出。现在已基本不用了。

点阵击打式印字设备是利用多根针经色带在纸上多次击打印出点阵字符的印字设备。当点阵字形构成的点较少时,虽字形可辨,但不清晰美观。随着技术的进步,打印头可以做到 48 针,点阵字形质量显著提高,几乎可与整字模印出的质量媲美。

点阵串行击打印字设备简称为**针式打印机**,俗称打印机。它的打印头是击打式打印机的成字部件。不同的成字机理,不同的打印头型式,就形成了各种

类型的打印机。我国广泛应用的针式打印机的打印头有 9, 14, 24 或 48 根针。针的直径为 0.2 ~ 0.3 mm。

打印头的基本工作原理是采用一定的动力源驱动打印针撞击打印媒体(色带与打印纸)获得印点阵而形成所需的文字、数字、符号或图像。

按驱动打印针动力源的不同形式,针式打印头有电磁式与压电式两类。

电磁式打印头利用电磁铁驱动打印针而获得印点。按电磁铁的类型,电磁式针式打印头分为螺管式、拍合式与储能式三种,分别如图 1(a), (b), (c) 所示。螺管式与拍合式同属吸合式,即线圈通电后,衔铁被吸合而驱动打印针撞击打印媒体。储能式打印头则利用永久磁铁的磁场与线圈通电时产生的反向磁场的相互作用使永磁吸力减小,吸力小于簧片弹性恢复力时,簧片释放使打印针撞击打印媒体而获得印点。螺管式打印头虽然稳定性好,噪声较低,但高速性差,故通常用于低档的 9 针打印机中。拍合式打印头高速性较螺管式的好,且其经济性、可靠性等均属优良,在 24 针打印机中应用很广。储能式打印头驱动电流小,功耗低,工作频率高,且可靠性高,易于小型化,其高速性、稳定性、低噪声等性能均较上述两者好。因此,在高档、高速、高可靠打印机中有广泛的应用。

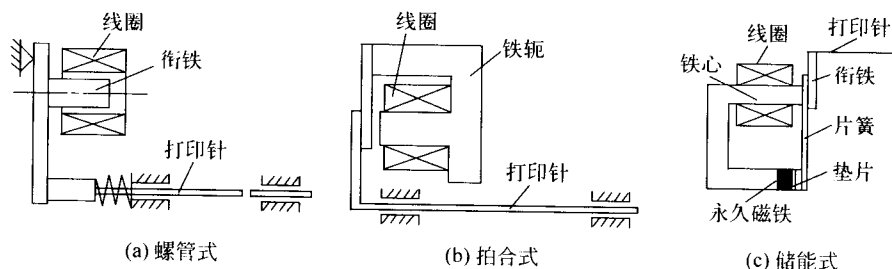


图 1 电磁式针式打印头

压电式打印头的工作原理是利用压电陶瓷材料的逆压电效应(压电晶体在外电场作用下成比例地产生几何形变的现象)驱动打印针撞击打印媒体而

获得印点。压电式打印头有两种形式:纵向效应型与横向效应型。纵向效应型采用较多,它利用压电材料平行于外电场方向的伸缩变形以驱动打印针。

压电材料是三元系钙钛矿型的铌镍酸铅系压电陶瓷(如 NEPEC-10)。由于压电材料的压电常数极小(NEPEC-10 的仅约为 $6.35 \times 10^{-10} \text{ m/V}$),因此,要使打印针得到约 0.4 mm 的位移量,必须加设扩大机构和采用多层薄片叠层的压电元件结构。

压电式打印头的打印速度高(达 140 汉字每秒),功耗小(约为电磁式的 1/2),体积小,重量轻,便于小型化,且噪声小,价格低,并有很高的寿命(达 10^{11} 次)。

打印机印字的过程是:印字头沿纸面自左向右按步移动,每步一个点距。驱动印针的电磁铁是受字符图形的点阵内容控制的。根据字符点阵的需要,一列印针分别击打或不击打。每完成一列点的打印,印字头右移一个点距。如此一列一列地打印,当印字头横向扫描至右端时,完成一行字符的打印。然后,印字头返回到起始位置,纸前进一个行距。有些打印机具有双向打印的功能,利用印字头的返回过程,在微型计算机的控制下,自右向左逐列打印,以便提高打印速度。

打印机的优点是可使用普通纸,并可同时复印数份。打印机的机构简单可靠,印字速度较快,字型变化灵活,耗材成本低廉。但其噪声大,打印质量差,逐渐地将被打印质量高、噪声低的喷墨印刷机和激光印刷机取代。尽管如此,但由于击打式票据打印机能够在多层纸和厚质材料(如很厚的证券卡或银行存折)上打印,因此票据打印机已广泛地用于处理商务发票、银行存折、报告单据、柜员收据、运输提单以及打印火车、公路、航空、轮船客票等方面。

点阵并行击打印字设备采用多针横行排列的梳状印字头,通常用多个打印头(如 12 个打印头)并排击打。(2003 年)汉字打印速度达到每分钟 450 行,针寿命高达 10 亿次以上。它应用于大批量高速持续打印的场合,如系统的数据备份。

参考文献

陈其昌,谢仕聘. 汉字打印机设计原理. 武汉:华中理工大学出版社,1995 (谢仕聘)

jiqu fanyi

机器翻译(machine translation, MT) 在计算语言学理论指导下,应用计算机技术,实现从一种自然语言(源语言)翻译成另一种自然语言(目标语言)的过程、方法和技术。又称自动翻译。

机器翻译是数字计算机在非数值领域应用的最早尝试。从 20 世纪 50 年代到 60 年代前期,美国曾

致力于机器翻译的研究,但由于低估了自然语言的复杂性及自然语言处理的理论与技术尚不成熟,机器翻译研究未能达到预期目标。1966 年美国发表了题为“语言与机器:翻译中的计算机与语言学”的报告(通常称之为 ALPAC 报告),它认为“假定机器翻译是指借助于一种算法把机器可读的源语言文本转换为有用的目标语言文本,而无需人来翻译或编辑,那么,在这个意义上,一般科技文章的机器翻译是不存在的,而且在最近的将来也不会有”,这对当时处于热潮中的机器翻译泼了一瓢冷水。其后 10 余年间是机器翻译研究的沉寂期。从 20 世纪 70 年代后期开始,由于计算机技术的发展和语言学理论的进步,特别是由于社会日益信息化对自动翻译的需求,在世界范围内,机器翻译研究与实用系统的开发再次得到蓬勃发展。中国也是世界上最早开展机器翻译研究的国家之一,曾于 1959 年进行了俄汉翻译的试验。经过 40 余年的发展,机器翻译在国内外都取得了很大的进展,在某些限定领域发挥了实际作用,并有不同水平的产品进入市场。但是,由于自然语言固有的复杂性及计算机的智能尚未充分发展等多种原因,机器翻译要在通用领域真正实现高质量和全自动,还需要攻克一系列的理论与技术难题。现在机器翻译研究已成为在高新技术领域进行竞争的热点之一。机器翻译研究丰富了计算语言学与人工智能的理论,有利于推进对人类语言理解机制的探索,从而为最终解开人类智能本质之谜做出贡献。

计算机进行翻译的必要条件是它能“懂”源语言并会“写”目标语言。

只要计算机系统中存放一部英汉对照的词典,就可以在词汇一级实现英语和汉语的互译。限于词汇的翻译满足不了人们的需求。更大的语言单位(句子或篇章)之间的自动翻译有更高的实用价值。不过,任何一种自然语言的句子都是无限集,只能利用有限资源的特定的计算机系统,无论如何不可能将两种语言的所有句子都以一一对应的形式存入系统。迄今为止,可以将源语言的任意句子作为处理对象的机器翻译系统的实现方法大致可分为两类:基于规则的方法和基于统计的方法。

基于规则的方法又分为直接方式、转换方式与中间语言方式。这 3 种方式都是建立在要素合成原理的基础之上的,即将源语言的句子分解为基本成分(如单词),利用各个基本成分之间的关系以及它们所对应的译词作为构成要素,重新构造目标语言的句子。

直接方式 早期的机器翻译系统采用这种方式。它从源语言的表层句子出发,将源语言的句子分解成单词,然后去查对译词典,并直接生成目标语言。这种方法以词汇转换为中心任务,也进行必要的词法分析和词序调整,语法知识混在算法与处理程序之中。这种方式主要用于句法结构相近的两种语言之间的翻译,能够处理的句型有限。

转换方式 多数实用型机器翻译系统基本上采用这种方式。它将分析源语言的表层句子所得到的内部结构转换为目标语言的内部结构,再由目标语言的内部结构生成目标语言的表层句子。表面上看,自然语言的句子只是由词排列成的线性序列,实际上句子是有结构的。英语的句子是由(名词短语+动词短语)组成的。名词短语由(名词)或(冠词+名词)或(冠词+形容词+名词)组成。动词短语由(不及物动词)或(及物动词+名词短语)组成。可用下列句法规则表述英语句子的这种结构:

$$S \rightarrow NP + VP$$

$$NP \rightarrow [det] + [a] + n$$

$$VP \rightarrow vi$$

$$VP \rightarrow vt + NP$$

其中,S,NP,VP 是非终极符,分别代表句子、名词短语、动词短语;det, a, n, vi, vt 是终极符,分别代表冠词、形容词、名词、不及物动词、及物动词。方括号中的语言成分是可选的。利用词典,可以查出终极符所代表的具体的词。利用这些规则,剖析下面的例句:

例 1. The girl likes red skirts.

可以得到表示其结构的树形图,见图 1。

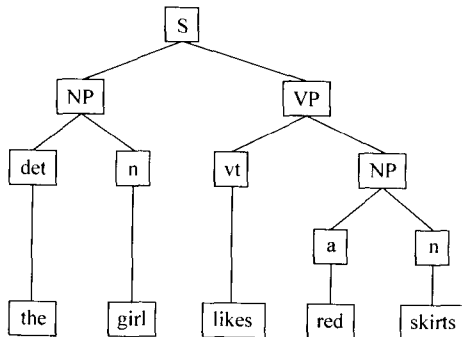


图 1 句子的树形结构

显然,一种句子结构可以代表数目众多的表层句子。一种自然语言有很多种不同的句子结构。如果只将覆盖面约束在满足一定实用要求的合理范围

内,句子结构的数量是可以控制的。而且,对于不同语言,表达等价意义的句子结构之间存在着对应关系。实际上,上述 4 条规则也反映了汉语的一类句子的结构,只不过将英语的定冠词需置换为汉语的指示代词。

计算机程序执行翻译任务时,就是按照句法规则把源语言的一个句子逐步分解,直至基本的构成要素,并找出这些构成要素之间的关系,这样便得到源语言句子的结构。这一步骤称为分析。源语言句子的分析过程还可以细分为:①词法分析;②句法分析;③语义分析;④语境分析。

词法分析又称形态分析。英语的词有形态变化,需要根据变化规则(包括名词复数、动词单数第三人称、动词的过去时与过去分词等)将源语言句子中的词形还原成基本形,这样才便于查词典。汉语词法分析的主要任务是词的自动切分和词性标注。

句法分析的任务是根据某种句法理论,建立一套句法规则(上述的几条规则是例子),系统据此对源语言的句子进行分析,得到其句法结构的机器内部表示(树形图)。在确定句子结构时,会遇到难以决断的歧义问题。

例 2. I bought a table with three dollars.

例 3. I bought a table with three legs.

在例 2 和例 3 中,最后面的介词短语可能是谓语的状语,也可能是前面名词的定语。要消解这样的歧义,需要依靠系统知识库中储存的语义知识进行语义分析。根据语义(常识)可以判断 with three dollars 是修饰 bought 的,而 with three legs 是修饰 table 的。语义分析通常在句法分析的基础上进行。同样,语义分析也不能消解所有句子的歧义。

例 4. She found a young girl with a telescope.

要确定例 4 中的 with a telescope 是修饰 found 的还是修饰 a young girl 的,只好根据上下文的动态知识进行语境分析,它是继语义分析之后的最后一个分析步骤。语境分析需要强有力的推理机制。

上面按词法分析、句法分析、语义分析、语境分析的顺序介绍了机器翻译系统中最主要的分析模块的流程(图 2 中的③,④,⑤,⑥)。

转换在分析所得的机器内部表示的基础上进行(图 2 中的⑦)。该内部表示可以通过句法分析、语义分析和语境分析得到(可能默认某些步骤)。

目标语言生成(图 2 中的⑧,⑨,⑩)是机器翻

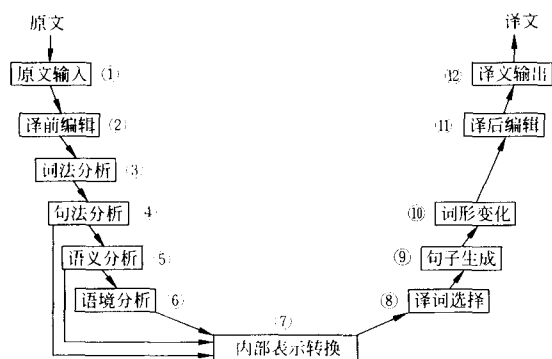


图2 机器翻译系统的流程图

译系统的最后一个主要模块。它根据转换模块所得到的目标语言句子的内部结构,按照一定的算法将位于这种结构中的通过查词典得到的目标语言的词语进行线性排列,从而得到目标语言的表层的句子。

中间语言方式 中间语言是一种不依赖于特定的自然语言的形式语言。它能够表达任何一种自然语言的表层句子的词汇所代表的概念及句子所表示的概念间的关系。采用中间语言方式的机器翻译系统分析源语言,将它变换为既不依赖于源语言也不依赖于目标语言的中间语言形式,然后由中间语言直接生成目标语言。为了开发多语种互译的机器翻译系统,采用中间语言方式在软件开发上是经济的,只需要开发 n 个分析程序和 n 个生成程序就够了。不过,理想的中间语言的设计非常困难,而且中间语言如何表示各种语言所特有的诸如时态、语态、语气之类的表层信息还是一个正在探索的课题。

基于大规模双语语料库的统计机器翻译方法有多种。其中基于实例的方法是最容易理解的。预先准备好足够多的每句源语言和每句目标语言都对齐的语料库。当要翻译源语言的句子 A 时,就从源语言句子库中找相似的句子 A' (最好找到相同的,通常不可能),然后根据语料库中与 A' 对齐的目标语言句子 B' 整体地生成句子 A 的译文 B 。最佳匹配检索技术和适当的调整机制是实现这种方法的关键。随着计算机技术的进步和大规模双语语料库的建成,基于统计的机器翻译方法将显示出它的生命力。

基于规则的和基于统计的两种方法的结合应当显示出更强的优势。

参考文献

1. 赵铁军等. 机器翻译原理. 哈尔滨: 哈尔滨

工业大学出版社, 2000

2. 冯志伟. 自然语言机器翻译新论. 北京: 语文出版社, 1994

3. W. J. Hutchins. 机器翻译: 过去、现在、未来. 台北: 致文有限公司, 1993 (俞士汶)

jiqi fanyi pingjia

机器翻译评价 (machine translation evaluation)

对机器翻译系统性能及其译文质量的评价。机器翻译评价对机器翻译(MT)的研究开发和推广应用有举足轻重的影响。机器翻译评价也是机器翻译领域的一个重要研究课题。

机器翻译的研究开发者,将机器翻译系统作为工具的翻译公司,要求得到最终译文的用户以及机器翻译评价的专门研究者都在做机器翻译的评价工作。不同人的评价角度与内容是不同的,得到的结论也可能大相径庭。必须综合不同类型人的观点,进行多方位的评价,才有可能得到比较全面、比较客观的结论。机器翻译的应用目标也是多种多样的,应用于信息检索系统的同应用于翻译技术说明书的显然应有不同的性能指标。实际上凡适于应用目标的就有生命力。统一的生产率观点有可能协调各种不同的评价,也即将翻译工作置于语言文字信息处理的全过程中加以考察,这个过程包括检索、识别、输入、译前编辑、翻译、译后编辑、输出、排版、印刷、远程通信等等,考察用机器代替人进行翻译是否提高了生产率。

机器翻译系统的性能包括翻译速度、人机界面、译文质量等诸多方面。只有适当的性能价格比,才可能赢得用户。

翻译速度通常用单位时间翻译的词汇量来进行计算。不过翻译速度与源语言文本的难易、译文质量要求又有关系。作为一个计算机系统,批处理方式的吞吐量与会话方式的响应时间也是重要的性能指标。

人机界面是决定机器翻译系统优劣的一个重要性能指标。用户可能选择译文质量较差但便于修改的系统,而不愿使用译文质量虽然好一些但修改起来甚不方便的系统,不能对译文进行编辑修改的旅游翻译器的实际用途是有局限性的。

关于机器翻译的译文质量的评价是机器翻译评价中最重要的也是最有特色的内容。译文质量评价对来说也是一项十分棘手的任务,“仁者见仁,智者见智”可以说是普遍现象。编制一个适当的测试

例句集是译文质量评价,甚至也是其他性能指标(如翻译速度)评测的基础。测试集既独立于具体的机器翻译系统,又独立于具体的测试方法。测试集除了包含源语言例句外,也需要附上正确的至少是可供参考的译文。

目前译文质量的评测工作基本上还是由人来做。半自动的比较测试是通过度量机器翻译的译文与正确译文之间的差异来实现的。两者之间的差异是指使机器翻译译文达到正确译文水平所需要的修改次数(如词的替换、调序、增删等)。修改工作是要人来做,但修改次数的统计与差异值的计算则是由机器自动进行的。

参考文献

1. 冯志伟,杨平. 自动翻译. 上海: 知识出版社, 1987

2. 陈肇雄主编. 机器翻译研究进展. 北京: 电子工业出版社, 1992 (俞士汶)

jiqiren chuanganqi

机器人传感器(robot sensor) 用于测量机器人及周边环境状态信息的装置。主要分为两类:一类是内部状态传感器,另一类是外部状态传感器。

内部状态传感器主要用来测量机器人的内部状态,如机器人关节变量、关节速度变量等。最典型的如安装在电机轴上的光码盘,它用来测量机器人的关节角位移。外部状态传感器主要用来测量机器人与外界环境之间相互关系的变量信息,如力觉、触觉、视觉、滑觉、接近觉等。机器人传感器主要指的是外部状态传感器。由于机器人视觉已发展成为相对独立的研究领域,因此若不特别说明,机器人传感器则主要是指非视觉传感器,即力觉、滑觉、接近觉等。

力和力矩传感器 用于机械手的力传感器,根据其所安放的位置可以分为以下三种:放在机械手的关节处,主要测量机械手的关节力矩;放在被操作物体的台架上;放在机械手的末端处,测量机械手末端的作用力。

(1) 关节力传感器 在上述三种力传感器形式中,关节力传感器是应用得最早的一种。早期主要用在主从机械手中。通过测量从机械手的关节力矩,并将其反馈到主手,从而使操作人员操纵主手时有力的感觉。对于电动机械手,关节力矩可以直接通过测量驱动电机的电流来获得。由于关节力矩的测量分布于整个机械手,因此它不仅可以检测手部

所受到的力,同时也能敏感测出机械手其他部分所受的力。该方法通过换算可以获得机械手末端的作用力,但精度较差。

(2) 台架力传感器 若将力测量装置安放在机械手所要操作的台架上,机械手的末端力也可以通过这种方法测量出来。它主要适用于利用机械手进行研磨、切削等工作。该方法的主要缺点是灵活性较差,它只适用于机械手从事某一特定工作的情况。

(3) 腕力传感器 力传感器安装在机械手的手腕处,故称腕力传感器。它直接测量机械手的末端的作用力,精度较高,是目前应用得较多的一种。腕力传感器通常要求体积小、重量轻和灵敏度高。腕力传感器的典型结构是在一个十字花梁的4根挠曲棒上粘贴8对半导体应变片,棒端的每一面均贴有一片。棒端对面的两边应变片连到差动电位器电路,其输出电压与垂直应变片平面方向的力成正比。差动连接可补偿温度变化的影响。8对应变片所产生的8组输出读数,根据力和力矩的平衡条件,解算出沿坐标轴方向的3个力分量和3个力矩分量。

触觉传感器 触觉传感器主要用来测量机械手与所操作物体相接触的有关信息,该接触信息可用于物体的定位和识别,也可用来控制机械手对物体所施加的力。触觉传感器主要分为两大类:二位信号传感器和连续信号传感器。二位信号触觉传感器实际是由如微动开关那样的接触装置所组成,它一般安装于手指的内表面,其作用不仅用于感知手指中间是否有物体存在,而且经过多次触摸,可以帮助机械手确定抓取位置。若二位信号传感器安装在机械手的外表面时,它可为机械手提供有用的导引信号。连续信号接触传感器是一种柔顺装置,它的输出正比于局部力的大小。常用的方法有:弹性杆测量,主要用来测量一个点的接触力;触觉传感阵列,可提供较大范围的触觉信息,通常装于手指的内表面;人工皮肤,通过感知接触变形来获得局部的力信号。

上述接触传感器测量的均为法线方向的力信号,如果测量的是切向力,则它可用作滑觉传感器。

接近觉传感器 接近觉传感器主要用来感知在一定的近距离范围内是否有物体存在,输出的是二位信号。接近觉传感器主要用于机器人在近距离范围的物体抓取和避碰,它通常有以下几种实现方法。

(1) 感应传感器 其工作原理是,当附近存在金属物体时,将引起电感的变化,由此可测知附近物体的存在。

(2) 霍尔效应传感器 它利用霍尔效应原理来感知附近物体的存在,它只适用于铁磁材料物体,其应用受到一定限制。

(3) 电容传感器 当附近存在物体时,将引起电容的变化,从而感知附近物体的存在。

(4) 超声传感器 它通过检测反射的声波来感知周围物体的存在,不受物体材料性质的限制。

(5) 光学传感器 它通过检测反射光来感知周围物体的存在。

距离传感器 它用来测量从参考点(通常是传感器本身)到物体间的距离。距离传感器主要用于机器人的导航和避碰,通常有以下几种实现方法。

(1) 三角测量法 发射光源发射光到附近的物体,改变发射角直至接收器接收到反射的光束。通过三角关系可算出到物体的距离。

(2) 结构光方法 通过发射出一定的结构光模式(常用窄光面)到附近的物体,物体上显示出一条光带,摄取该光带信息并送至计算机分析即可获得距离信息。

(3) 激光测距 通过测量发射与接收光之间的时间间隔来获取距离信息。

(4) 超声测距 通过测量发射与接收声波之间的时间间隔来获取距离信息。

机器人传感器是传感器技术在机器人中的具体应用,它是机器人感知外界信息的输入接口,是实现机器人智能化的必要工具和手段。目前机器人传感器正朝着高性能、小型化、集成化和智能化的方向发展。

参考文献

Fu K S, Gonzalez R C, Lee C S G. Robotics: Control, Sensing, Vision and Intelligence. McGraw-Hill Book Company, 1987

(孙增圻)

jiqiren kongzhi

机器人控制 (robot control) 研究以机器人为对象的控制问题的领域。

机器人的动力学模型具有强耦合和非线性的特点,是一个难于控制的复杂对象。机器人控制的主要问题在于研究性能优良、易于实现的控制方法。目前,实用的机器人主要采用独立关节的 PID 伺服控制,它具有稳定性好、控制简单的特点。但当机器人的运动速度较高时,该方法的缺点便明显暴露出来。人们从不同角度对机器人的控制问题进行了深入的研究,并提出了各种不同的方法,比较典型的

有:分解运动速度控制、分解运动加速度控制、计算力矩控制、具有滑动模态的变结构控制、非线性控制、自适应控制等。

机器人的运动控制通常划分为两个阶段。第一阶段是从起始位置到达目标点附近沿期望轨迹的粗略运动,这时主要要求位置的控制;第二阶段是精细的运动控制,这时机器人的末端执行器对物体进行操作,以完成要求的任务,同时需要增加外部力觉反馈的柔顺运动控制。它不同于纯粹的位置控制,而是机器人控制中的另外一个重要内容。机器人柔顺控制主要采用阻抗控制和位置、力混合的控制方法。

由于对机器人的精确建模比较困难,尤其是当考虑摩擦等实际的非线性因素时更是如此。因此,人们对于不太依赖模型的机器人智能控制方法日益感兴趣。其中较为突出的是机器人模糊控制和神经元网络控制,它们越来越受到人们的关注和重视,并已显示出广泛的应用前景。

机器人控制方法的研究始于 20 世纪 70 年代。80 年代,对各种控制方法的研究达到高潮,这些方法多数处于研究阶段,真正实际应用的并不是很多。其主要原因是它对模型的要求比较高,或者算法过于复杂,难于实时实现。目前,机器人控制更多地集中于基于多传感器信息融合的智能控制方法的研究。

目前,机器人控制的工程实现与一般控制相类似,也主要采用计算机控制的结构形式,以实现更为复杂和更高要求的控制。

参考文献

1. Fu K S, Gonzalez R C, Lee C S G. Robotics: Control, Sensing, Vision, and Intelligence. McGraw-Hill Book Company, 1987

2. 孙增圻,严隽薇,钱宗华. 机器人智能控制. 太原: 山西教育出版社, 1994

(孙增圻)

jiqiren shijue

机器人视觉 (robotic vision) 模拟人的视觉使机器人系统具有感知环境和工作对象之能力的技术。它能有效地增强机器人对环境的适应能力和扩大应用范围。

机器人视觉的应用领域有以下几方面: ①为机器人的动作控制提供视觉反馈。其功能为识别工件,确定工件的位置和方向以及为机器人的运动轨迹的自适应控制提供视觉反馈。需要应用机器人视觉的操作包括:从传送带或送料箱中选取工件、制造

过程中对工件或工具的管理和控制,例如,使焊枪沿相对于边缘或其特征的预定路径移动以及有感知反馈的装配操作。②移动式机器人的视觉导航。这时机器人视觉的功能是利用视觉信息跟踪路径,检测障碍物以及识别路标或环境,以确定机器人所在方位。③代替或帮助人工对质量控制、安全检查进行所需要的视觉检验。

机器人视觉的基本要求是进行实时处理,即在机器人操作过程中同时完成视觉信息的处理。然而进行灰度图像处理和获取三维信息需要完成大量的计算,许多情况下难以实时完成。室内环境下的机器人视觉,由于可对工作环境作较严格的控制,因此可利用合适的约束来简化机器人视觉的计算。例如,利用特殊的光照或选用特殊的传送带颜色使得通过图像的二值化处理就能把工件与背景相分离。所以,初期的机器人视觉大多数利用二值化图像,抽取工件图像的面积、矩、细长比等二维特性来完成工件的识别和定位。二值化图像的缺点是难以提供三维信息,而有些机器人操作必须有深度信息的导引。为了简化处理,在机器人视觉中经常采用人工光照,例如用激光结构光的方法来获取深度信息。结构光的方法避免了双目立体视觉处理中的复杂计算,便于实时处理。

近年来由于研制能在特殊或恶劣环境下工作的室外移动式机器人的需要,对机器人视觉的研究提出了全新的要求,如要求处理彩色图像,获取三维信息,实时处理并且能适应室外环境多变的光照。这就使机器人视觉成为机器视觉的重要研究内容。

参考文献

1. Robert M Haralick and Linda G Shapiro. Computer and Robot Vision. Addison-Wesley Publishing Company, 1993
2. Charles E Thorpe, ed. Vision and Navigation. The Carnegie Mellon Navalab, Kluwer Academic Publisher, 1990
3. Batchelor B G, Hill D A and Hodgson D C, ed. Automated Visual Inspection. IFS (Publications) Ltd., 1985

(徐光佑)

jiqiren yundong gui Hua

机器人运动规划 (robot motion planning)

在满足一定的物理约束条件下,对机器人到达预期位置及取向的动作序列所作的安排。

对于智能机器人领域中最常见的研究对象——

机械手和移动机器人来说,所要满足的物理约束条件主要是避免与环境中的其他物体发生碰撞。在这种意义下,机器人运动规划问题可描述为:令 A 为欧氏空间 W (工作空间) 中运动的一个刚性物体——机器人; B_1, \dots, B_q 为分布于 W 中的固定(或运动)的刚体,称为障碍;假设 A, B_1, \dots, B_q 的几何参数以及 B_i 在 W 中的位置是精确已知的,同时还假设 A 的运动不受任何动力学的约束,给定 A 在 W 中的初始状态和目标状态(位置和方位),要求产生一个路径 P , 它确定一个使 A 从初始状态开始,于目标状态结束,且不与 B_i 发生碰撞的运动状态的连续序列。如果这种路径存在的话,则给出可行解或在某种意义上下的最优解。

机器人的运动规划是体现机器人智能水平的重要标志,也一直是智能机器人研究领域的热点和难点。规划器、传感器和执行器一起被认为是智能机器人的三大核心单元。运动规划的研究涉及到人工智能、理论计算机科学、数学、机械工程等多学科领域。由于决定机器人运动规律的逆运动学是一个多维多值的非线性函数,而机器人和障碍物通常是具有复杂几何形状及多自由度的三维物体,因而对运动规划的求解提出了较高的要求,也增加了算法的复杂性。这方面的典型结论包括:①三维多面体环境中任何 L^p 度量下寻找最短路的问题是 NP-HARD; ②二维环境中具有运动障碍(若运动物体的速度有限)的运动规划是 NP-HARD。

机器人运动规划的早期研究可以追溯到 60 年代末期。Howden 于 1968 年发表题为“The sofa problem”的文章,其中的“sofa”指的是一个二维物体 A , 环境为一个复杂的二维结构 M , 所要解决的问题是物体 A 能否在 M 内从一点 P_i 运动到另一点 P_k 。虽然 Howden 对运动规划问题的讨论作了诸多简化,但其中的思想和算法却蕴含了后来对规划问题所发展出的各种技术的基本思想,特别是关于运动规划问题的一般求解结构:①划分空间;②建立包含划分信息的网络;③在网络上搜索路径。

20 世纪 80 年代出现了大量的具有重要理论和实际意义的研究工作。Lozano-Perez 系统地提出了利用姿态空间来求解机器人无碰路径的方法。其基本思想是将机器人的位置及方向变换为姿态空间中的一点,该点的每个坐标表示机器人的位置或方向的一个自由度,相应地将环境中的障碍物也映射到姿态空间中,从而使机器人在工作空间中寻找路径变换成一个点在姿态空间中寻找路径。其他的典型

方法包括 Freeway 算法、可视图方法、Subdivision 算法、收缩法、人工势场法。除基于几何的方法外,还有基于拓扑的方法以及运动规划算法的复杂性讨论。

针对该领域尚存在的主要问题,近期研究的新进展包括:①针对高维空间运动规划实时性差的缺陷,考虑快速在线的运动规划方法研究;②针对基本运动规划问题中结构化静态环境、鲁棒性较差等局限性,重点考虑在感知空间中研究运动规划问题,如基于传感器的感知行为和反射式规划的研究,机器学习与传感器自组织等问题的研究,具有运动障碍物的运动规划问题研究;③针对环境模型和机器人运动控制的不确定性,研究分析和处理环境模型误差和机器人的控制误差等。

参考文献

1. Howden W E. The Sofa Problem. Computer Journal, Nov. ,1968,11:299 ~ 301
2. Lozano-Perez T. Spatial Planning: A Configuration Space Approach. IEEE, Trans. on Computers, Feb. ,1983,32:108 ~ 120
3. Chien R T, Zhang L and Zhang B. Planning Collision - free Paths for Robotic Arm Among Obstacles. IEEE, Trans. on PAMI,1984,6:91 ~ 96

(王宏)

jiqi xuexi

机器学习 (machine learning) 计算机模拟或实现人类的学习行为,以获取新的知识或技能,或者重新组织已有的知识结构使之不断改善自身的性能的过程、原理和方法。它是人工智能领域的一个重要分支。

学习能力是智能行为的一个非常重要的特征,但至今对学习的机理尚不清楚。人们曾对机器学习给出各种定义。H. A. Simon 认为,学习是系统所作的适应性变化,使得系统在下次完成同样或类似的任务时更为有效。R. S. Michalski 认为,学习是构造或修改对于所经历事物的表示。从事专家系统研制的人们则认为学习是知识的获取。这些观点各有侧重,第一种观点强调学习的外部行为效果。第二种则强调学习的内部过程,而第三种主要是从知识工程的实用性角度出发的。

机器学习在人工智能的研究中具有十分重要的地位。一个不具有学习能力的智能系统难以称得上是一个真正的智能系统,但是以往的智能系统都普遍缺少学习的能力。例如,它们遇到错误时不能自

我校正;不会通过经验改善自身的性能;不会自动获取和发现所需要的知识;它们的推理仅限于演绎而缺少归纳,因此至多只能证明已存在的事实、定理,而不能发现新的定理、定律和规则等。随着人工智能的深入发展,这些局限性表现得愈加突出。正是在这种情形下,机器学习逐渐成为人工智能研究的核心之一。它的应用已遍及人工智能的各个分支,如专家系统、自动推理、自然语言理解、模式识别、计算机视觉、智能机器人等领域。其中尤其典型的是专家系统中的知识获取瓶颈问题,人们一直在努力试图采用机器学习的方法加以克服。

机器学习的研究是根据生理学、认知科学等对人类学习机理的了解,建立人类学习过程的计算模型或认知模型;发展各种学习理论和学习方法,研究通用的学习算法并进行理论上的分析;建立面向任务的具有特定应用的学习系统。这些研究目标相互影响相互促进。

自1956年人工智能奠基以来,人们非常重视机器学习的研究,至今已经历了四个发展阶段。

第一阶段始于20世纪50年代中期,这一阶段的工作不少程度上受启发于神经生理学、生物学等研究,主要是研究不需要什么初始知识的通用学习系统,特别是神经网络系统。这一阶段的一个重要特点是数值表示和参数调整,不像当时的人工智能领域重心在于符号表示和启发式方法,而更偏向于模式识别。这一时期的代表性工作有感知机、生物进化过程模拟,以及 A. L. Samuel 的很有名的、曾击败过州级冠军的计算机跳棋学习程序。

第二阶段始于60年代初期,这一阶段的不少工作受到心理学和人类学习的启示,主要是概念学习和语言获取,有人称其为符号概念获取阶段。这一时期的主要特点是使用符号表示而不是数值表示。当时符号表示已成为人工智能的主要方法。另外,采用数值表示的神经网络由于 M. Minsky 等于1969年发表的著作从理论上分析了感知机的能力和限制,使得这方面的研究陷入低谷。这一时期的代表性工作有概念学习系统 CLS,积木世界结构学习系统。另外,在学习计算理论方面,建立了极限辨识理论。

第三阶段始于70年代中后期,由于专家系统的成功和知识工程的形成,这一阶段的工作对知识的重要性尤其关注。一方面,领域知识大量引入学习程序之中;另一方面,知识的自动获取成为机器学习的应用目标。这一时期,机器学习逐渐走向兴盛,各

种学习策略、学习方法相继出现,除了作为主流的归纳学习外,还出现了类比学习、解释学习、观察和发现学习等等。这一时期有影响的工作有学习质谱仪预测规则系统 Meta-DENDRAL,利用 AQ 11 方法学习大豆疾病诊断规则系统,利用 ID 3 方法学习象棋残局规则,数学概念发现系统 AM,符号积分系统 LEX,以及一系列物理定理重新发现系统 BACON。在学习计算理论上,L. G. Valiant 提出了概率近似正确 PAC 学习模型,这一成果推动了学习计算理论的发展。

第四阶段始于 80 年代中后期,主要源于神经网络的重新兴起。由于使用隐单元的多层神经网络及反传算法的提出,克服了早期线性感知机的局限性,从而使得非符号的神经网络的研究得以与符号学习并行发展。同时,机器学习在符号学习的各个方面也更加深入和广泛地展开,并形成了较为稳定的几种学习风范,如归纳学习、分析学习(特别是解释学习和类比学习)、遗传学习等。这一时期有影响的工作有多层神经网络反向传播学习算法,基于解释的学习,一系列决策树归纳学习方法,J. H. Holland 的遗传学习和分类器系统,A. Newell 等的 SOAR 学习系统,以及 PRODIGY 学习系统等。近期,由于复杂世界的实际应用的需要,出现了结合各种学习方法的集成学习系统、多策略学习技术,特别是关于连接学习与符号学习的结合。另外,有着很大应用价值的数据库知识发现学习技术也发展得很快。

机器学习经过三十多年的发展,到现在已形成了很多学习方法,例如机械学习、传授学习、实例学习、发现学习、解释学习、类比学习、事例学习、遗传学习、连接学习等。这些学习方法可以用一个学习模型来描述(参见图 1)。

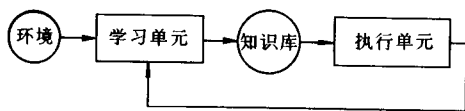


图 1 一个简单学习系统模型

在图 1 中,圆圈表示信息体(如观察的数据,以及事实、规则等知识),方框表示过程。箭头指示数据在学习系统中的流向。环境为学习单元提供外界信息源(如经验实例)。学习单元利用该信息对知识库作出改进(增加新知识或重新组织已有知识)。执行单元利用知识库中的知识执行任务,任务执行后的信息又反馈给学习单元作为进一步学习的输入。

学习单元的输入有两种:一是外界环境,另一是执行任务后的反馈信息。不同的学习系统有不同的经验实例表示。最简单的一种是二元特征表示,仅仅描述对象某些属性的存在与否,例如病人有或没有某个特定症状。下文要讲的连接学习和遗传学习方法一般使用这种二元特征的输入。另一种是用属性值表示,每个属性有一组相互排斥的值,如颜色属性的值可为红色、蓝色和黄色等。二元特征可看作是此类的特例。这种属性值表示典型地用在归纳学习方法中。还有一种更复杂的表示是关系或结构表示,描述两个或多个对象间的关系,如对象 A 位于对象 B 的上方。这种关系或结构信息一般是以谓词逻辑、语义网络等形式表示的。同前两种表示相比,这种表示具有更强的表示能力,但同时也为作为学习中重要部分的匹配过程带来了相当的复杂度,以至影响了它们的使用。分析学习主要处理这种关系表示型的数据结构。

知识库用来存储知识,包括系统原有的领域知识(这种知识是长期的、相对稳定不变的),以及通过学习而获得的各种新知识(这种知识是短期的、相对不稳定、变化的)。选择何种知识表示对学习系统的设计起着非常重要的作用,如是存储具体的单个的经验实例还是仅存储从这些实例得到的抽象推广。机器学习的大部分方法是只存放总结了以前经验的普遍性的知识,而分析学习中的类比学习及基于范例的学习则是混合了两种方式。如果知识是后一种表示(以抽象概括的形式存储),则存在两种区别,即是以逻辑的、离散的形式表示,还是以数值的、连续的形式表示信息。归纳学习和分析学习使用的是离散的、逻辑的表示方法,连接学习则使用数值的、连续的表示,遗传学习则是两种表示的结合。

执行单元既是使得学习系统具有实际用途,又是评价学习算法性能好坏的关键部分。机器学习研究的很大一部分工作是集中在这样两个领域:分类和问题求解。一个智能主体常常需要将其经验或经历的事例进行分类标记。例如,在面临病人表现的某些特定症状时,医生需要作出某一疾病的诊断。机器学习至今所作的很大一部分是学习分类。该任务可描述为,给定一个经验实例的某种描述,以及一些已知的类,任务在于试图将这些经验实例赋予某个类别。专家系统所作的最常见的也是最有用的工作便是同这种分类或诊断有关。另外,机器学习也常要能解决新问题或进行规划,例如,任务是设计一个从当前所在地到达目的地的路径。一般是给定一

些所要达到的状态或目标,任务在于试图找出一个动作序列,使得其主体能从当前状态达到目标状态。机器人操作是最明显的问题求解策略的应用。这个领域涉及技能的获得,用以改进问题求解的速度和规划推理的能力。

学习单元是学习系统内实现学习算法功能的核心。一般涉及这样几个方面,一种是处理经验事例的方式,有渐进式和非渐进式。渐进式是指每次仅处理一个事例,能不断地处理新遇到的事例。因而,这种方式能处理很大的(理论上是无穷大的)数据集。非渐进式则指仅用一次时间处理有相当大小的一个事例集,这种方式的长处是能根据大量数据的统计特点获得很多决策、推断信息。

另一种是获取新知识过程中所用的推理方法,主要有归纳和演绎方法,也有反绎方法。归纳方法主要基于观察数据来形成一般化的知识,它的特点在于产生的知识是先前知识库中所没有的(或不能蕴涵的)。它主要用于归纳学习、连接学习和遗传学习等学习风范。演绎方法则依赖于知识库中已有的知识来形成新知识,如基于解释的学习是利用先前的知识来解释新的事例,然后简化该解释并存放于知识库中。这种方法主要用于分析学习中。另外,归纳和演绎方法的结合,比如类比学习能得到更好的学习效果。

当考虑执行单元의 反馈信息时,还存在两种相对的学习方式,即**监督学习**和**非监督学习**。前者是指有导师立即给予学习者(学习单元)关于其学习行为的反馈信息,后者则是学习者得自己为自己作出判定,或仅能得到一点很粗略的指导。在分类任务领域里,监督学习一般称为实例学习。在分类任务中,非监督学习的学习者必须自己决定自己的类以及类的数目。这种学习称为概念聚类。在问题求解领域中,监督学习指的是导师能在求解的每一步搜索过程中向学习者建议正确的操作或子目标。在这方面,研究得更多的是非监督学习,学习者得靠自己区分哪种动作是自己所希望的,这种方法称作信任赋值。

目前,机器学习风范主要有归纳学习、分析学习、遗传学习、连接学习等。

归纳学习从具体实例出发,通过归纳推理,得到新的概念或知识。归纳学习的基本操作是泛化和特化。泛化是使规则能匹配应用于更多的情形或实例。特化操作则相反,减少规则适用的范围或事例。归纳学习是目前研究得最广泛的一种符号学习方

法,包括实例学习、概念聚类、发现学习等。实例学习的任务是,给定关于某个概念(或多个概念)一系列的已知的正例和反例,要求从中归纳出一个(或多个)一般的概念描述,该描述能使得这些已知的正例可从中再次推导出来而同时没有任何反例可从中推导出来。概念聚类则是由程序根据实例间的某种类似度关系自动形成有用的概念描述(参见**归纳学习**)。发现学习主要是从实验数据、观察实例或数据库中获得知识。

分析学习是利用背景或领域知识,分析很少的典型实例(通常仅一个),然后通过演绎推导,形成新的知识,使得对领域知识的应用更为有效。分析学习方法的目的在于改进系统的效率性能,而同时不牺牲其准确性和通用性,这不同于归纳学习方法。目前,常见的分析学习方法有解释学习、范例学习(参见**分析学习**)、类比学习(参见**类比学习**)等。

遗传学习源于模拟生物繁殖中的遗传变异原则(交换、突变等),以及达尔文的自然选择原则(生态圈中适者生存)。一个概念描述的各种变体或版本对应于一个物种的各个个体,这些概念描述的变体在发生突变和重组后,经过某种目标函数(相应于自然选择准则)的衡量,以决定谁被淘汰谁继续生存下去(参见**遗传算法**)。

连接学习是在人工神经网络中,通过样本训练,修改神经元间的连接强度,甚至神经网络本身的结构的一种学习方法。这种学习方法和符号学习并行而行。它与符号学习不同,主要是基于样本数据进行学习(参见**人工神经网络**)。

机器学习已获得了不少的应用,其中,应用最广泛的是归纳学习,人们已经将决策树方法用于工业过程控制,金融保险预测以及医疗诊断等领域。在连接学习方面,神经网络也在模式识别、智能控制等方面得到了应用。

机器学习已建立了一定的理论基础和自己的科学研究方法,形成了各种研究风范以及相应的学习算法。但是,目前的机器学习风范均有其局限性。在了解人类学习机制的基础上,正结合具体的应用领域,深入开展学习风范的研究,研究不同学习风范的集成,探讨新的学习方法和算法。

参考文献

1. Michalski R S, et al. eds. Machine Learning: An Artificial Intelligence Approach, Vol I, II, III. Morgan Kaufmann, 1983, 1986, 1990
2. Shi Zhongzhi. Principles of Machine Lear-

ning. Beijing: International Academic Publishers, 1992
(史忠植 王军)

jìqǐ yǔyán

机器语言 (machine language) 表示成数码形式的机器基本指令集, 或者操作码经过符号化的基本指令集。

机器语言一般由一台机器可以执行的全部指令及其所操作的数据组成, 其功能可以通过相应计算机的基本指令集合 (也称作指令系统) 加以描述。其中, 每条指令将指挥计算机执行一个基本操作, 包括数据处理操作 (如算术运算、逻辑运算、字符处理等), 控制操作 (如判断、转移、中断、改变机器状态等) 和传输操作 (如输入、输出、数据移动等)。在计算机中, 实施操作的指令和被实施操作的对象均要表示成二进制代码形式。指令由操作码和地址码两部分组成, 操作码指明要实施的基本操作; 而地址则指明被实施操作的对象在计算机中的存放位置。被实施操作的对象可以是整数、实数、布尔值、字符串等, 由于它们最终均要表示成二进制数字序列的形式存放在计算机中, 为了区分它们的类型, 通常要在操作码中设置“标志”字段来加以标识, 这与高级语言中利用类型说明来标识对象的类型是有所不同的。机器语言的主要特点是与特定的机器相关, 运行效率较高级语言高, 但用户难于使用, 繁琐, 费时, 且易出错。
(曹东启)

jìqǐ zhōuqī

机器周期 (machine cycle) 计算机在执行一条指令的过程中, 完成各种特定功能的时间段。

计算机执行一条指令的过程可以划分成若干功能阶段, 每个阶段完成一定的操作功能。一条典型指令的执行过程如下: 第一步是取指令, 从指令计数器中把要执行的指令的地址送入地址寄存器, 从存储器中取出指令, 送到指令寄存器; 第二步是分析指令, 对指令寄存器中的操作码进行译码, 确定指令应该完成的功能, 并对寻址方式进行解释, 确定操作数的有效地址; 第三步是取数, 根据操作数的有效地址, 取出操作数; 第四步是执行, 按照操作码的规定, 对操作数进行具体处理; 第五步是回送处理结果。以上各步的时间段分别叫做取指令周期、指令译码周期、取操作数周期、执行周期和回送结果周期。不同指令需要不同的机器周期, 例如有的指令

寻址方式是变址寻址或间接寻址, 为了求出操作数的有效地址, 在指令分析时还需增加变址周期或间址周期。为了进行 I/O 数据传输, 还需要有中断周期、DMA 周期等。

机器周期的时间宽度取决于完成该周期的功能所需的时间。不同机器周期所需的时间是不同的。为了设计简便, 在划分指令功能阶段时, 使完成每个阶段所需的时间差不多。具体设计时, 使各个机器周期宽度相同。因此机器周期的宽度应取决于各周期中所需时间的最长者。早期的一些计算机常将一个机器周期分为有一定时间顺序的若干节拍, 以实现机器周期内部各种操作的顺序控制要求。以取指令周期为例: 该周期的第 1 节拍将要执行的指令地址由指令计数器经地址总线送往内存地址寄存器, 第 2 节拍向内存发读出命令, 经过若干时间等待, 最后一个节拍将读出的指令经数据总线送到指令寄存器中。节拍电位的宽度一般决定于中央处理器执行一次运算或寄存器间完成一次数据传送所需时间, 通常与计算机的时钟周期一致。

不同机器周期所需要的节拍数是不同的。可以用以下方法决定机器周期的宽度: ①统一节拍法, 所有机器周期的节拍数都是一样的, 这样控制比较简单, 但对某些周期, 时间有些浪费。②可变节拍法, 允许多种机器周期的节拍数不同, 但控制复杂一些。③延长节拍法, 根据多数机器周期需要的节拍数, 作为其基本节拍数, 特殊情况可再延长一个节拍或两个节拍。④插入节拍法, 当某个节拍的操作不能完成时, 可插入若干个等待节拍。

设计高速计算机的主要目标是提高指令执行速度, 这要求减少每条指令中机器周期的数量和减少机器周期的时间宽度。如果一个机器周期的功能在一个时钟周期内完成, 也就是一个机器周期只含有一个时钟周期, 这时, 机器周期与时钟周期一致, 统称为周期。例如 Intel Pentium 的整数指令采用 5 级流水, 有 5 个机器周期, 即 PF 指令预取, D₁ 指令译码, D₂ 生成地址, EX 执行, WB 写回结果, 每个周期的操作都在一个时钟周期内完成。在流水线结构的情况下, 一条 5 级流水线可同时有 5 条指令重叠执行, 在任一时刻, 每条指令处于不同的功能阶段。每一个时钟周期启动一条指令, 每一个时钟周期完成一条指令。

机器周期的宽度对计算机的性能有很大影响, 除了改进微电子工艺以提高器件速度外, 还与指令系统设计有很大关系。例如采用 RISC 技术时, 指令

多为寄存器-寄存器型,这样可以减少访问内存操作。如果机器采用流水线结构,可让多条指令重叠执行。如果采用超标量结构,每个时钟周期可同时从指令高速缓存中读出多条指令并行操作,机器速度将有更大提高,例如 DEC Alpha, Super SPARC, Intel 80860 都是采用超标量结构。超流水线结构的每一级流水线分解为多级更短的流水线,例如 MIPS R4000 将原来的 4 级流水线变成 8 级流水线,在一个基本时钟周期内分时发出两条指令,从而使机器周期的宽度更进一步缩短。

参考文献

李三立, 李亚民. RISC 单发射与多发射体系结构. 北京: 清华大学出版社, 1993 (谢树煜)

ji zhu ren yi

机助人译 (machine-aided human translation, MAHT) 机器翻译的一种类型。翻译工作的主要部分由人完成,但其工作方式是计算机化的。翻译的对象是文件形式的源语言文本,翻译好的目标语言文本也以文件形式存入计算机,当然,在需要时也可以打印或排版输出。

由于机助人译的自动化程度不高,狭义的机器翻译不将它包括在内。但由于自动翻译的译文质量不能令人满意,机助人译仍有其实用价值。计算机程序即使仅按照源语言句子中的单词排列顺序查出目标语言的对应的译词,翻译者在双语对照的窗口环境中,在全屏幕编辑软件的支持下,翻译工作的效率也会提高很多。对于一词多义的情况,可以同时列出:系统配备一部有多种索引的惯用语词典,也会帮助翻译者解决不少难点。专业词典的效用更是显而易见的。如果系统中配备了大量的双语对照的译例,当人遇到不易处理的句子时,就可以查找这个译例库。翻译者在这样的翻译工作站上搞翻译工作,不仅提高了效率,而且自身的水平也会不断得到提高。

参考文献

冯志伟, 杨平. 自动翻译. 上海: 知识出版社, 1987 (俞士汶)

jishu

基数 (cardinal number) 有穷集合元素个数概念的推广和精确化。

若存在从集合 A 到集合 B 的双射,则称 A 与 B

对等,记为 $A \sim B$ 。与自然数集合 \mathbf{N} 对等的集合称为可数集。集合 A 为无穷集当且仅当 A 对等于它的某个真子集。由所有与 A 对等的集合组成的集合类,称为 A 的基数,用 \bar{A} 或 $\#A$ 表示。

空集的基数规定为“0”,非空有穷集合的基数就是它的元素个数,并称它们为有穷基数。无穷集合的基数称为**超穷基数**或**无穷基数**。

自然数集合 \mathbf{N} 的基数用 \aleph_0 表示,实数区间 $(0, 1]$ 称为**连续统**,它的基数用 \aleph 表示。G. Cantor 首先证明 $\aleph_0 \neq \aleph$ 。因为从 $\mathbf{N} \times \mathbf{N}$ 到 \mathbf{N} 有如下的双射 π :

$$\pi(n, m) = \frac{(n+m)(n+m+1)}{2} + n \quad n, m \in \mathbf{N}$$

所以 $\overline{\mathbf{N} \times \mathbf{N}} = \bar{\mathbf{N}} = \aleph_0$,即 $\mathbf{N} \times \mathbf{N}$ 为可数集。

设 A 和 B 为任意两个集合。若有 $B' \subseteq B$ 使 $A \sim B'$,则称 A 的基数小于等于 B 的基数,记为 $\bar{A} \leq \bar{B}$ 。如果 $\bar{A} \leq \bar{B}$ 且 $\bar{B} \neq \bar{A}$,则称 A 的基数小于 B 的基数,记为 $\bar{A} < \bar{B}$ 。

定理 (伯恩斯坦定理) 设 A 和 B 为两集合。若 $\bar{A} \leq \bar{B}$ 且 $\bar{B} \leq \bar{A}$,则 $\bar{A} = \bar{B}$ 。

定理 (康托尔定理) 若 A 为集合,则 $\bar{A} < \overline{\mathcal{P}(A)}$ 。

以上两定理的证明见参考文献 1。

设 A 和 B 为两个集合,用 B^A 表示集合 $\{f|f:A \rightarrow B\}$ 。

基数加法 若 $A \cap B = \emptyset$,则称 $\overline{A \cup B}$ 为 \bar{A} 与 \bar{B} 之和,记为 $\bar{A} + \bar{B}$ 。

基数乘法 称 $\overline{A \times B}$ 为 \bar{A} 与 \bar{B} 之积,记为 $\bar{A} \cdot \bar{B}$ 。

基数的幂 称 $\overline{B^A}$ 为 \bar{B} 的 \bar{A} 次幂,记为 $(\bar{B})^{\bar{A}}$ 。

有以下关于基数运算的重要公式:

(1) 若 $n \in \mathbf{N}$ 且 $n \neq 0$,则

$$\aleph_0 + n = \aleph_0 = n + \aleph_0$$

$$\aleph_0 + \aleph_0 = \aleph_0 = \aleph_0 \cdot \aleph_0$$

$$n \cdot \aleph_0 = \aleph_0 = \aleph_0 \cdot n$$

(2) 若 $n \in \mathbf{N}$ 且 $n \neq 0$,则

$$\aleph + n = \aleph = n + \aleph$$

$$\aleph + \aleph_0 = \aleph = \aleph_0 + \aleph$$

$$\aleph + \aleph = \aleph = \aleph \cdot \aleph$$

$$n \cdot \aleph = \aleph = \aleph \cdot n$$

$$\aleph_0 \cdot \aleph = \aleph = \aleph \cdot \aleph_0$$

(3) $\aleph = 2^{\aleph_0}$

(4) 若 α, β 和 γ 都是基数,则

$$\begin{cases} \alpha + \beta = \beta + \alpha \\ \alpha \cdot \beta = \beta \cdot \alpha \end{cases} \quad \text{交换律}$$

$$\begin{aligned}
 & \left\{ \begin{array}{l} (\alpha + \beta) + \gamma = \alpha + (\beta + \gamma) \\ (\alpha \cdot \beta) \cdot \gamma = \alpha \cdot (\beta \cdot \gamma) \end{array} \right. \quad \text{结合律} \\
 & \left\{ \begin{array}{l} \alpha \cdot (\beta + \gamma) = \alpha \cdot \beta + \alpha \cdot \gamma \\ (\alpha + \beta) \cdot \gamma = \alpha \cdot \gamma + \beta \cdot \gamma \end{array} \right. \quad \text{分配律} \\
 & \left\{ \begin{array}{l} \alpha^{\beta \cdot \gamma} = \alpha^{\beta} \cdot \alpha^{\gamma} \\ (\alpha \cdot \beta)^{\gamma} = \alpha^{\gamma} \cdot \beta^{\gamma} \\ (\alpha^{\beta})^{\gamma} = \alpha^{\beta \cdot \gamma} \end{array} \right. \quad \text{指数定律} \\
 & \left\{ \begin{array}{l} 0 \cdot \alpha = 0 = \alpha \cdot 0 \\ 0 + \alpha = \alpha = \alpha + 0 \\ \alpha^0 = 1 \end{array} \right. \\
 & \left\{ \begin{array}{l} 1 \cdot \alpha = \alpha = \alpha \cdot 1 \\ \alpha^1 = \alpha \\ 1^{\alpha} = 1 \end{array} \right.
 \end{aligned}$$

(5) 若 α 为超穷基数且 β 为非零基数, 则

$$\alpha + \beta = \max\{\alpha, \beta\}$$

$$\alpha \cdot \beta = \max\{\alpha, \beta\}$$

参考文献

王兵山, 王长英, 周贤林, 何自强. 离散数学. 长沙: 国防科技大学出版社, 1985 (王兵山)

jìyú Agent de jìsuan

基于 Agent 的计算 (agent-based computing)

在人工智能中, 由多个 Agent (智能体、主体或代理) 之间的合作、协作与协调, 通过建立模型进行问题求解的一种计算模式, 可以用于复杂分布问题的建模、设计和求解。智能 Agent 是处于特定环境中的计算机系统, 并能采取灵活、自治和理性的行为完成相应的目标或任务。一般认为 Agent 是具有自治性、智能性、反应性、预动性和社会性的计算实体。多 Agent 系统 (MAS) 是多个 Agent 合作求解的具体体现, Agent 之间通过交互协调各自的行为, 以实现个体或整体目标。MAS 具有自然的分布式求解机制, 多重控制线索, 多个问题描述角度。MAS 更能体现人类社会的智能, 具有更大的灵活性和适应性, 更适应动态开放问题的求解。MAS 有合作的和非合作的形式。合作的 Agent 有共同的求解目标, 主要研究目标分解、组织形成、求解协议以及如何保持 Agent 行动的一致性。非合作的 MAS 由自利的 Agent 组成, 每个 Agent 有自己的求解目标, 其中有些 Agent 可能是敌对关系, 也可能是资源竞争关系, 主要研究社会规范、交互协议、对手模型等。MAS 的研究涉及心理学、哲学、经济学、认知科学、管理学等多门学科。

基于 Agent 的计算的研究可以追溯到 20 世纪 70 年代, Minsky 在《思维的社会》中提出的 Agent, 认为社会中的某些个体经过协商可求得问题的解, 这些个体就是 Agent。

80 年代初, V. Lesser 开发了分布式车辆测试床 (DVMT), 研究了 Agent 组织和局部规划共享等问题; C. Hewitt 提出了 Actor 计算模型, Actor 是具有消息传送能力并可以分布控制的活动体, 可以在开放环境中通过并行交互进行求解; R. Smith 提出了合同网协议, 该协议是一个高层协议, 让有任务求解需求的 Agent 与有求解能力的 Agent 建立联系并形成求解合同。

80 年代后期, 研究集中在 MAS 规划和协商等方面。多 Agent 规划包括有中央规划器的多 Agent 规划、有统一协调机制的多 Agent 规划和完全分布式的多 Agent 部分规划。V. Lesser 和 E. Durfee 提出了部分全局规划模型, 多个 Agent 通过任务分配、局部规划和局部规划的交互与协调逐步形成部分全局规划。Rosenschein 研究了 Agent 在目标冲突情况下的交互问题, 运用对策论建立了理性 Agent 交互的静态模型, 阐述了 Agent 协作的条件。

90 年代, MAS 的研究更加活跃, 研究的内容也更加广泛, 包括 Agent 模型、Agent 规划与合作求解、Agent 通信语言和面向 Agent 的程序设计、基于对策论和其他经济学方法的分布式决策、Agent 联盟和组织、多 Agent 学习、以及 MAS 在各领域的应用等。用信念、愿望和意图等思维属性对 Agent 建模是 Agent 模型研究的主要工作。这期间 Rao 和 Georgeff 的 BDI 模型, Cohen 和 Levesque 的意图理论有较大影响。Agent 通信语言以言语行为理论为基础, 主要有 ARPA 提出的知识查询与操纵语言 (KQML) 和 FIPA 标准中的 Agent 通信语言 (ACL)。Y. Shoham 提出了面向 Agent 的程序设计的计算框架, 可以直接用思维属性进行 Agent 程序设计。此外, 多 Agent 拍卖机制、基于微观经济学市场平衡理论的多 Agent 决策和多 Agent 学习等问题的研究在这一时期都取得了一定进展。

目前基于 Agent 的计算研究主要包括 Agent 理论模型、Agent 结构、联盟和组织、Agent 语言、Agent 交互技术、多 Agent 学习、移动 Agent 以及 Agent 技术的应用等。

Agent 理论模型 建立理性 Agent 的形式化模型, 为 Agent 系统的建造奠定基础。目前研究的主要内容包括个体思维状态模型、群体思维状态模型

以及个体思维状态和群体思维状态的关系。Agent 思维状态模型借用心理学和哲学的研究成果从认知高度描述 Agent 的各种思维属性及其关系,以及它们对 Agent 规划、行为、协调、合作等活动的影响。主要结果有 BDI 模型,意图理论和承诺交互理论等。

Agent 结构 Agent 组成结构有慎思型结构、反应式结构和混合型结构。慎思型结构有显式的符号模型和推理能力,是一种基于知识的系统,包括对环境的认知、符号表示以及推理部分;反应式结构没有符号表示的现实世界模型,没有复杂的符号推理,系统根据条件-动作规则对环境的变化直接作出反应;混合型结构是前两种结构的结合,包含推理和反应两个子系统,分别用来进行规划决策和直接对环境事件作出反应。

联盟与组织 联盟的研究主要基于对策论方法,讨论联盟结构的形成,联盟值的计算和分配等问题。Agent 组织的研究更注重组织结构、组织形成和组织演化,以及各种基于 Agent 组织的求解机制和应用等问题。

Agent 语言 包括面向 Agent 的程序设计(AOP)和 Agent 通信语言(ACL),分别用于 Agent 系统的设计和 Agent 之间的通信。AOP 的关键思想是可以直接对 Agent 进行基于思维状态的编程,它包括语言定义、实现和工程化三个主要方面。ACL 的实现主要基于言语行为理论。

多 Agent 交互 交互是多 Agent 合作求解的核心问题之一,Agent 通过交互可以达成协作或协调各自的行动。交互方法包括基于协商的合同网协议、基于依赖关系的社会推理、基于价格调控的市场机制等。协商有基于对策论的协商和基于劝说的协商,协调分为显式协调和隐式协调。

多 Agent 学习 一方面多 Agent 在分布环境下通过交互与合作学习,改进合作和问题求解效率,以适应开放环境的动态变化;另一方面,通过多 Agent 学习的研究改进对自然系统和计算系统学习理论和学习机制的理解。

移动 Agent 移动 Agent 是一类特殊的 Agent,它除了具有智能 Agent 的基本特性外,还具有移动性。移动 Agent 计算模式能有效的降低分布式计算中的网络负载,提高通信效率,支持离线计算和异步自治交互,可动态适应网络环境,具有安全性和容错能力。移动 Agent 需要解决的关键技术问题主要有:移动、通信、程序设计语言、安全性、容错、管理、理论

模型和协作模型等。

Agent 技术的应用 基于 Agent 的计算有广阔的应用前景,主要用于电子商务和电子市场、实时通信检测与管理、运输调度与交通管理、互联网信息处理、日程安排、工业制造与生产过程优化、娱乐游戏以及社会系统仿真等。

参考文献

1. Nicholas R Jennings. Agent-Based Computing: Promise and Perils. In Proceedings of IJCAI-1999, 1429 ~ 1436
2. Gerhard Weiss. Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press, 2000
3. Wooldridge M 著. 多 Agent 系统引论. 石纯一等译. 北京: 电子工业出版社, 2003

(石纯一 张伟)

jìyú ànlì de tuīlǐ

基于案例的推理 (case-based reasoning, CBR) 依据过去求解类似问题所积累的案例或经验来求解新问题的一种推理模式,它是人工智能领域中的一种重要推理技术。

CBR 研究起源于认知科学,其思想最早由 R. Schank 在 1982 年出版的《动态记忆》(Dynamic Memory)中提出,1983 年 J. Kolodner 等人开始在计算机上实现 CBR 系统。进入 20 世纪 90 年代, CBR 受到越来越多的关注,并得到进一步深入广泛的研究。

同产生式或其他基于规则的推理系统相比, CBR 系统以一种完全不同的方式来解决新问题。在 CBR 系统中,领域知识由一组按一定方式存放在案例库中的案例描述,这些案例记录了过去相关问题的解法。CBR 在求解问题时,首先对案例库进行检索,找出过去解决相似问题的案例,利用它来解决新问题;原案例解可能不完全适合新问题,还需对其进行检验和修改;最后生成新问题的解,同时也得到新的案例,并存入案例库,表示系统学到了新的知识。因此 CBR 的一个推理周期可以分为四个过程,即检索、重用、修改和保留。检索过程从案例库中找出与当前问题最相似的案例;重用过程试图用相似案例方法求解当前问题,产生建议解;修改过程在案例的建议解不适合时,对其进行修改;保留过程将已证实的解作为新案例存入案例库。

CBR 方法与基于规则的方法相比具有如下明显的优点: ① CBR 不需要详尽的领域知识模型,知

识获取变为逐步积累求解案例,是解决基于规则的系统知识获取瓶颈的一条途径;②系统实现的主要问题变为对案例特征的提取和表示,这较建立一个详尽的领域模型要容易;③储存的大量案例可利用先进的数据库技术进行管理;④CBR系统通过积累案例来学习新知识、改善性能,系统易维护。

由于CBR方法比较符合人类记忆和推理的思维方式,因而是一种具有吸引力和有发展前景的问题求解方法。目前已有一些基于CBR的系统问世,并被应用于预测、诊断、规划、设计、软件工程等方面。进一步开拓CBR的应用领域,结合其他人工智能技术和计算机技术提高CBR系统的智能程度和求解效率是CBR未来的主要研究方向。

(赵沁平 王军玲)

jiyu goujian de ruanjian kaifa fangfa

基于构件的软件开发方法 (component-based software development method, CBSD) 一种基于预先开发好的软件构件,通过将其集成组装的方式来开发软件系统的方法。又称基于构件的软件工程(CBSE),它是软件复用的实现方式之一。其根本目的仍然是为了提高软件开发的质量和效率。

基于构件的软件开发方法(CBSD)的兴起主要是源于如下不同的背景:一是在学术研究方面对现代软件工程思想,特别是对软件复用技术的高度重视;二是在技术研发方面所取得的有效进展,如,虽然缺少理论的支持,但在图形用户界面(GUI)和数据库应用中基于部件的组装技术的成功应用;三是一些主流互操作技术开发者的积极推动,如OMG的CORBA/CCM、微软公司的COM/DCOM以及SUN公司的EJB已成为主流的构件实现规范,相应的软件中间件平台规范也已获得较为普遍接受;四是由于面向对象技术的广泛使用,提供了构件和使用构件的概念基础和实用工具,事实上,主流的构件实现模型均基于对象技术。

从开发方法的角度来看,CBSD提供了一种自底向上的、基于预先定制包装好的类属元素(构件)来构件应用系统的途径。应该看到,CBSD的发展和中间件技术的发展是密切相关的,正是中间件技术及其平台提供了构件开发和构件组装的技术基础和机制。因此,当前CBSD讨论的重点主要局限于基于COM/DCOM、CORBA/CCM和EJB等主流规范

的二进制级构件。

从复用的角度看,CBSD支持的是黑盒、组装式复用方式。系统开发者不能对构件进行源代码级的修改,最多只能是通过参数方式进行适应性调整。构件的组装在中间件平台上进行,构件间通过中间件提供的通信协议和设施进行交互。

CBSD的主要活动包括:构件获取、构件认证、构件适应性修改、构件组装和应用部署。

构件获取 根据应用需求,去寻找符合或基本符合需求的构件,通常途径有三,一是来自第三方构件开发商,二是货架商品式的(COTS)构件,三是集成者根据特殊应用需求开发。

构件认证 对候选构件进行有用性、可用性、可信性和复用历史等方面的考察,以确定是否可用于新应用系统中。认证可由集成者自己完成,也可委托第三方认证。

构件适应性修改 由于应用的特殊性,有时需要对候选构件进行适当修改。由于CBSD所支持的黑盒复用模式,集成者只能通过参数调整的方式修改构件。如要涉及代码级的修改,通常仍需由原构件开发商完成。

构件组装 根据应用的功能及非功能需求,将相关的构件组装在一起,主要工作涉及一些胶合代码和构件间连接关系描述信息的生成。

应用部署 将构成应用系统的各构件或构件组分别部署到相应的运行环境中,也即是说将组装形成的应用系统“安装”到相应中间件平台上。部署完毕,应用系统方可投入运行。

基于构件的软件开发方法(CBSD)正在受到越来越多的关注,然而,CBSD的有效性和高效性还面临一些挑战。CBSD当前主要着重于目标码层次的互操作能力,缺少涉及需求和设计层次的系统化方法学支持,因此,需要和软件复用的已有研究成果进行无缝整合。同时,大量可供使用的构件的存在是CBSD能够被广泛应用的前提,而这正是目前大多数应用领域所欠缺的。此外,多种构件实现规范及体系结构风格的共存,使得软件体系结构失配成为CBSD方法的一个难题。

参考文献

1. Heineman G T and Councill W T. Component-Based Software Engineering: Putting the Pieces Together. Reading, Massachusetts, Addison-Wesley, 2001
2. Mili H, Mili A, Yacoub S. Reuse Based Software Engineering: Techniques, Organizations, and

Controls. John Wiley & Sons, 2002

3. Hong Mei, Jichuan Chang, Fuqing Yang. Software Component Composition based on ADL and Middleware. Science in China (F), 2001, 44(2): 136 ~ 151
(谢涛 梅宏)

jiyu heyi de yufa lilun

基于合一的语法理论 (unification-based grammars)

采用复杂特征集来描写词汇和短语的句法—语义信息,并在分析过程中采用合一运算作为归约或演绎基础的语法理论。与此相对的是只采用 NP(名词短语)、VP(动词短语)、PP(介词短语)等单一标记的短语结构语法。20 世纪 80 年代以来陆续提出了一批基于合一的语法理论,如词汇功能语法(LFG),功能合一语法(FUG),广义短语结构语法(GPSG),和定子句语法(DCG)等。这些形式各异的语法理论有以下两个共同特点:①拒绝采用 Noam Chomsky 关于浅层结构和深层结构的语言观,无需在分析或生成中沿用转换规则,因而又称为“非转换的语法理论”;②从词汇层开始就采用复杂特征集来描写词条的词法、句法和语义信息,并在分析过程中普遍运用合一运算作为归约或演绎的基础,所以得名为“基于合一的语法理论”。

定义 α 为一复杂特征集,当且仅当 α 可用如下形式来表达:

$$\begin{bmatrix} f_1 = v_1 \\ f_2 = v_2 \\ \vdots \\ f_n = v_n \end{bmatrix}$$

其中 $n \geq 1$,且特征名 f_i 为原子,特征值 v_i 既可以是原子,也可以是另一个复杂特征集。

式 $\alpha(f_i) = v_i$ 读作集 α 中的特征 f_i 的值等于 v_i 。

定义 合一

(1) 若 a, b 均为原子,则 $\overline{a \cup b} = a$,当且仅当 $a = b$;否则 $\overline{a \cup b} = \emptyset$ (其中, \cup 为合一运算符, \emptyset 为空集,用来表示合一失败)。

(2) 若 α, β 均为复杂特征集,则①若 $\alpha(f) = v$,而 $\beta(f)$ 未定义,则 $f = v$ 属于 $\alpha \cup \beta$;②若 $\beta(f) = v$,而 $\alpha(f)$ 未定义,则 $f = v$ 属于 $\alpha \cup \beta$;③若 $\alpha(f) = v, \beta(f) = v'$,则 $f = (v \cup v')$ 属于 $\alpha \cup \beta$;否则 $\alpha \cup \beta = \emptyset$ 。

从合一定义的(2)③来看,这是一个递归定义。

由于 f 的值 v 和 v' 本身又可以是复杂特征集,因此仅当 v 和 v' 可以合一时, α 和 β 才能合一,否则合一失败。

如果把自然语言看作是一个信息传递系统,并且承认语言的合成性假设,即无论是句法成分还是语义成分都是按由小到大的方式逐步组合起来的,那么采用合一作为句法和语义分析的基本运算是理想的。因为:①一个句子或其他语言单位所承载的信息总是分布在它的各个组成成分之中,因此每个成分所承载的只是局部信息;②通过合一把小成分组合成较大成分的过程中,每个小成分所承载的信息被传递或累加成为较大成分所承载的信息,在此过程中信息只增不减;③由于句法和语义信息可以同时进行传递和累加,因此不仅某些句法结构的歧义可以通过语义信息来排除,而且在句子的句法结构和语义表示之间找到了一种更自然的衔接方式;④合一运算是无序的,即不论合一从哪个方向开始,不论运算的先后次序如何,合一结果都一样。这一特性不仅适宜于并行处理,而且使我们有可能自由地选择不同的语法模型。

参考文献

1. 石纯一,黄昌宁,王家庶. 人工智能原理. 北京:清华大学出版社,1993

2. Shieber S M. An Introduction to Unification-Based Approaches to Grammar. CSLI Lecture Notes 1986,(4)
(黄昌宁)

jiyu leixing lilun de fangfa

基于类型理论的方法 (type theory based method)

使用类型理论的概念和方法解释程序设计中的问题并指导程序设计的方法。它是一种用于设计和实现软件开发环境和工具的形式化方法。

类型理论起源于 20 世纪初人们对数学与逻辑的基础研究。人们发现对数学对象的定义和某些逻辑规则的任意使用都会导致悖论。这迫使人们对这些数学对象和规则进行一定的限制,由此产生了构造性数学及直觉主义逻辑。其基本思想是:从直观的、可机械把握的简单对象和规则出发,使用构造性方法将数学对象和理论定义为不同类型,严格规定它们之间的关系、组合原则和使用规则。这种作法可以避免悖论的出现。D. Brouwer, Russell, 及 Church 等在这方面都做过巨大贡献。

70年代以来 P. Martin-Löf 等人将这种思想引入计算机科学,建立了以软件与程序设计为直接应用背景的类型理论。其中有代表性的是直觉主义类型论、爱丁堡逻辑框架及高阶构造演算等。现代类型论的特点是将古典类型论的可构造性强化为计算机可执行的算法或能行可计算的方法,以使相应的类型理论在计算机上可实现。基于类型理论的方法基本要点可以概括为以下步骤:

(1) 根据处理的问题的不同,选择一种类型理论,并根据这种类型理论,设计和实现一种相应的程序语言。这种语言应包含丰富的类型,能描述类型的构造和类型间的转换,并有快速而有效的类型检查功能。

(2) 就像数学理论可以用集合论的概念和方法描述一样,把待编程的问题用类型理论描述。例如,把“程序功能描述”看成一个类型,而实现此描述的程序,根据类型理论,就是这个类型的元素;把一个人工智能问题看成一个类型,则此问题的解就是这个类型的元素等。

(3) 根据类型理论提供的构造性方法进行程序设计,以解决这个问题。基于类型理论的方法的优点是可以将过去在软件工程与人工智能中所涉及的高层次“抽象概念”,如:“程序描述”,“规则”,“问题”,“求解策略”等进行形式化描述,把它们定义为某些类型,而与之对应的“程序”、“具体的程序变换规则”和“问题的算法”就成为相应类型的元素。根据类型理论这些“程序”、“变换规则”和“求解问题的算法”正确与否,就可以归结为类型检查问题,可由计算机辅助完成。这种作法可以避免软件开发中高层概念和工具的使用错误,提高软件开发效率。

参考文献

1. Harper R, Honsell F and Plotkin G. A Framework of Defining Logic, Proc. Symposium on Logic in Computer Science, IEEE Computer Society, June, 1987

2. Martin-Löf P. Intuitionistic Type Theory, Studies in Proof Theory, Lecture Notes, BIBLIOPOLIS, Napoli, 1984

(李未)

jìyú nèiróng de jiǎnsuǒ

基于内容的检索 (content-based retrieval)

抽取每个媒体对象的特征和语义并建立索引,通过搜索和匹配来实现多媒体对象检索的方法。基于内容的检索是针对传统的基于关键字检索在用于多媒

体对象时的不足提出来的。多媒体对象可以是文本、图像、视频、音频等数据。传统的基于关键字的检索方法往往需要对每个媒体对象(文本除外)配一个简要的文字说明,然后通过对说明文本的检索来寻找媒体对象。说明文件一般需要人工建立,这是一个非常耗时而繁琐的工作,同一个媒体对象可能由于用户或任务的不同而有完全不同的说明文件。也就是说它往往不能正确表达媒体的内容。基于内容的检索首先对多媒体数据进行分析,从中抽取每个媒体对象的特征和语义并建立索引。这些工作尽可能地让机器自动完成。检索的过程是近似匹配的过程,这种近似匹配可以在三个抽象层次上进行:

(1) 原始数据层 最低的抽象层。媒体对象被看作是像素或信号的简单聚集,媒体对象之间的比较通过像素或信号之间的相似度计算来进行。该层的检索需要较精确的匹配。

(2) 特征层 特征是指媒体对象具有区分功能的主要特性或属性,如图像中的颜色、纹理、形状,视频中的镜头、场景、镜头的运动,声音中的音调、响度、音色等。媒体对象不同的特征通常可组合成特征组,在多媒体数据库中用来匹配和搜索。

(3) 语义层 最高抽象层。在这一层的检索中,假定特征已经被归类成有意义的对象,检索基于良好定义的时空特性。例如某个建筑、某段情节、某段旋律等。语义层的检索通常是领域相关的,需要用户的较多参与。

基于内容的检索技术广泛应用于遥感图像处理、空间探测、医疗图像、建筑工程图、生物学、天气预报和艺术馆藏资料管理等许多领域。随着 Internet 的发展,视频、音频、图形和图像数据在网上大量涌现,基于内容的检索技术已成为不可缺少的检索手段。

(陆玉昌)

jìyú quǎnshì hánshù de huìzhì

基于全视函数的绘制 (plenoptic function based rendering) 根据全视函数生成景物全景图像的一类基于图像的绘制方法。

全视函数是由 Adelson 和 Bergen 命名的,用来表示空间任意点在任意时刻、任意波长范围内接收的全部光线的集合,它描述了给定场景中所有可能的环境映射。Adelson 和 Bergen 将全视函数定义在一个七维的参数空间上:

$$\mu = \text{Plenoptic}(\theta, \phi, \lambda, V_x, V_y, V_z, t)$$

其中, (V_x, V_y, V_z) 代表空间中视点的位置, 仰角 ϕ 和方位角 θ 表示视域方向和范围 (图 1), λ 代表人眼感受到的波长。如果是动态场景, t 代表时间。

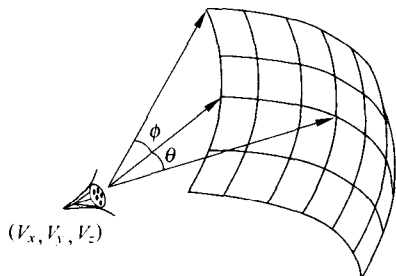


图 1 全视函数的参数化

基于全视函数的绘制方法试图捕获空间任意区域内的完全光流以重建该函数, 这一思想固然很好, 但实践中要在每一时刻、每一方向对场景的所有点进行度量极其不便。因此, 现有的方法都是通过一些有限的离散采样来重构连续的全视函数 P , 然后通过在新的视点位置重新取样该函数来计算新的视图。而且, 同时考虑所有 7 个参数比较困难, 故目前所有的方法都通过增加约束来简化全视函数, 如固定视点、固定时间, 或者约束环境等。典型的方法如基于五维全视函数的 PlenOptic Modeling 系统, 基于四维全视函数的 Light Field 系统, 基于三维全视函数的同心拼图等。

例如, 如果只考虑视点处水平 360° 的环境映射, 就构成了柱面全景图。PlenOptic Modeling 就是一个柱面全景图系统, 它利用立体视觉技术来寻找全景图间的对应匹配点, 并通过插值合成新的柱面全景图。美国苹果电脑公司的 QuickTime VR 也是面向柱面全景图的 IBMR 系统, 它是第一个商业产品。

光场绘制基于自由空间中沿一条光线传递的辐射能不变的假设, 它把图像作为二维的图像片插入一个四维的全视函数中, 解决了没有遮挡的、光照固定的、静态对象的表示及绘制问题, 但没有完全解决虚拟环境漫游问题, 只适合小场景的描述。

同心圆拼图把全视函数由四维降为三维函数, 其基本思想是把相机位置约束在一组平面同心圆上, 然后将沿每个圆周移动相机所得到的一系列图像拼合成一组同心拼图。同心圆拼图把全视函数参数化为 3 个参数: 半径、旋转角和垂直仰角。与固定视点的柱面全景图像不同, 同心拼图获得的全景图

视点不必是固定的, 它允许用户在一个圆内任意走动, 并能明显观察到景物图像的视差和光线的变化。新视图的绘制只要适当组合同心拼图中捕获的光线信息即可, 不需要恢复任何几何和光测定信息。

参考文献

1. Adelson E H and Bergen J R. The Plenoptic Function and the Elements of Early Vision. In: M Landy and J Movshon, editors, Computational Models of visual Processing, chapter 1. MIT Press, Cambridge, MA, 1991
2. McMillan L, G Bishop. Plenoptic Modeling: An Image-Based Rendering System. In: Computer Graphics (SIGGRAPH '95 Conference Proceedings), August 6 ~ 11, 1995. 39 ~ 46
3. Shum H Y and He L W. Rendering with Concentric Mosaics. In: Proc. of SIGGRAPH '99, Los Angeles, California, 8 ~ 13 August, 1999 (徐丹)

jìyú túxiàng de huìzhì

基于图像的绘制 (image-based rendering, IBR) 利用预先获得的一组图像 (通常是真实世界的照片), 通过一定的变换操作, 把它们组合成不同视线方向和 (或) 不同视点的新视图的过程和技术。基于图像的绘制与基于几何的计算机图形绘制方法不同。首先, 基于图像的绘制 (IBR) 不依赖于几何模型, 而是利用一组真实场景照片来生成场景的新视图, 故能反映极为丰富的景物细节和色彩, 有效地克服了计算机图形学存在的建模复杂度高和绘制真实感不足的缺陷。其次, IBR 的绘制速度不依赖于场景复杂度, 只取决于图像的分辨率, 有效地克服了计算机图形学绘制速度随场景模型复杂度增加而下降的缺陷。基于图像的绘制技术为实时绘制极为复杂的真实场景提供了可能。

基于图像的绘制技术始于 20 世纪 90 年代初。有关 IBR 的研究内容包括: ①数据采样方法, 指如何获得场景图像, 如采用不同的设备、拍摄方案和后期处理方式等; ②数据表示方式, 指如何看待采样的场景数据, 如可以把采样图像看作场景中光线的集合 (全视函数表示)、狭缝图像的集合、分层图像的集合或常规图像等; ③数据压缩和传输, IBR 的特点是计算量小, 数据量大 (数百帧高质量的图像), 如何有效地压缩, 并能快速转换为需要的表示方式进行显示是一项重要的研究内容。在短短的十年时间里 IBR 技术取得快速发展, 主要有基于全视

函数的绘制、视图插值和视图过渡等技术。

参考文献

1. Chen S E. QuickTime VR—An Image-based Approach to Virtual Environment Navigation. In: Computer Graphics (SIGGRAPH '95 Conference Proceedings), 1995. 29 ~ 38

2. 石教英主编. 虚拟现实及实用算法. 北京: 科学出版社, 2002 (石教英)

jiyu tuxiang de zaoxing

基于图像的造型 (image-based modeling,

IBM) 对实际景物进行拍摄, 根据获得的一组图像来重建其三维几何和外观模型的过程和技术。传统计算机图形学采用几何表示和纹理合成来建立对象或场景的模型, 称为基于几何造型技术。建立一个复杂对象或场景的三维几何模型是一项既困难, 又十分耗时的工作, 迄今仍缺乏有效的解析方法来综合复杂的外观特征。基于图像的造型技术直接用包含丰富细节和逼真外观特征的采样图像来造型, 改善了模型的复杂度和真实感。同时, 基于图像的造型技术的造型过程自动化程度高, 降低了造型成本。基于图像的造型技术是计算机图形学与计算机视觉两个学科交叉的产物, 在 20 世纪 90 年代中期兴起, 立即受到学术界重视, 成为研究、开发和应用的热点。

基于图像的造型技术分两类: 一类称为主动技术, 另一类称为被动技术。主动技术采用人为控制照明的方法进行拍摄, 以获得具有特殊效果的图像。例如将具有已知结构的光 (常用的有光栅条纹, 或仅仅一条棒形阴影) 投射到对象上进行拍摄, 根据光栅条纹在图像上的扭曲形状来恢复对象表面的三维模型。被动技术在拍摄时对光照条件不加任何限制, 具有较高的灵活性, 完全根据对象自身结构来重建模型, 其重建技术相对比较复杂。被动技术包括多种方法, 它们之间的主要区别在于对照相机校准的要求不同, 及重建过程中人机交互量的多少 (即自动化程度的高低) 两个方面。常用的被动技术有以下几种方法:

(1) 基于立体视觉的图像造型方法 假如用符合针孔模型的照相机在不同视点上拍摄对象, 得到多幅图像。在多幅图像上提取一系列特征点, 并找出它们的匹配关系, 据此进行相机自标定, 恢复相机的内外参数。然后将两幅相关图像重投影到某个公共平台上, 以使对应极线对齐, 再通过配准得到对应

的立体像对。根据立体像对与相关视线关系恢复像素点的相对深度, 达到三维重建的目标。

(2) 基于可见轮廓的图像造型方法 假设从不同视角拍摄同一个物体, 得到一系列图像。从每幅图像上可以得到从不同视角看到的物体的轮廓。物体的轮廓是理解物体几何形状的一个重要线索。从每个视点发出经过各自轮廓点的射线分别构成一系列锥形外壳。这些不同角度的锥壳的交集构成了物体的可见外壳 (三维模型)。该方法特别适用于带高光的、透明的、半透明的和带绒毛的物体的造型。

(3) 基于表面光场的图像造型方法 光场是指某一点上朝给定方向发出的光线。这样, 图像上的像素可以看作是物体表面光场中朝视点方向发出的光线集合。该方法对给定物体的上半空间 (假定下半空间为不可见) 进行全方位密集采样, 得到成百上千帧图像。这些图像记录了物体在上半空间的表面光场数据。当要求再现上半空间某视角方向上物体的外观时, 只要对物体的表面光场数据按给定的视点坐标和视线方向进行重采样, 即从表面光场选取符合条件的光线构成物体在给定视角上的外观图像。所以这是一种用于获取物体外观模型的方法, 不能获取物体的几何模型。由于该方法采样数据量极大, 因此有效的光场数据压缩技术成为其关键技术。

有关基于图像的造型的主要研究内容为: ①采样方法研究; ②采样数据的紧凑表示 (压缩技术); ③有效的绘制方法等。

参考文献

石教英主编. 虚拟现实及实用算法. 北京: 科学出版社, 2002 (石教英)

jiyu wuli de donghua

基于物理的动画 (physically based anima-

tion) 将物理特性引入模型, 并允许对模型的行为进行数值模拟的计算机动画技术。它可应用物理定律及基于约束的技术来推导和计算物体随时间运动和变化的状况。

这种技术的特点如下:

(1) 采用基于物理的造型, 使模型中不仅包含几何信息, 也包含物理信息。它将与行为有关的物理特性 (如质量、力、力矩、惯量等)、形体间的约束关系及其他与行为的数值模拟相关的信息引入模型中, 甚至将图像绘制技术也与该模型的描述相结合。

这样就模糊了通常的造型、动画、图像绘制之间的界限,改变了计算机动画通常采用的造型—动画—图像绘制相分离的流水线机制。同时,在实现形体的动态变化时也不再需要由人工详细规定或调整形体的几何数据和拓扑结构。

(2) 在形体的运动和变化的控制过程中引进了物理定律。用这种方法控制的运动更符合实际情况,更加自然也更有吸引力。

真实感效果是计算机动画所追求的目标之一。它不仅体现在“照相逼真”上,而且还体现在动画形体的造型及其动态变化和运动的逼真模拟上。基于物理的动画正是为解决这一问题而提出的。

计算机动画中要解决的另一个问题是如何正确地、容易地控制动画中形体的运动和变化。传统的**关键帧动画**在控制层次上偏低,需要由动画设计者直接控制形体的位置等相关参数,这对复杂的运动来说是相当繁琐和困难的,有时甚至是不可能的。

基于物理的动画采用基于物理模型的模拟技术,使动画设计者通过对形体及影响形体动作的环境随时间而变化的描述来实现控制,这就提供了很大方便,即使是初学者也可自动地生成真实的运动。基于物理的动画已发展成计算机动画中的重要课题。它与机器人学、人工智能、面向对象的方法相结合,适用于通用的集成环境中,实现任务级上的运动控制和描述。如能有效地解决运动和变化的计算速度问题,必将在**科学计算可视化**、**视觉模拟**、**人体动画**等领域发挥更大的作用。

参考文献

1. Barzel R. Physically Based Modeling for Computer Graphics: A Structured Approach. Academic Press, 1992

2. Foley J D, Dam A V, et al. Computer Graphics: Principles and Practice. Addison Wesley, 1990

(王裕国)

jìyú wùlì de zàoxíng

基于物理的造型 (physically-based modeling) 在几何造型的基础上,将形体的物理性质引入其模型,并对其行为和形状的变化进行数值模拟的造型技术。

许多物体的行为和形状是由该物体与其他物体的相互关系及有关的物理性质所决定的。例如,覆盖在物体上的布的形状是由表面摩擦力、构成布的

织物的质量及来自物体的力所产生的应力和应变所决定的;悬浮在两根柱上的链条呈弧状下垂,它是由重力及维持链条间的连接力所决定的。基于物理的造型就是解决这类物体(可以是刚体、弹性体、塑性体,也可以是人、动物、机器人的关节体等)的具有物理真实性的动态造型问题。它与传统的造型技术不同,在建造模型时,不仅包含几何信息,也引入了导致物体形状和行为变化所需要的其他信息,即物体的物理性质和数值模拟机制。这些信息涉及运动学(位置、速度等)、动力学(力、力矩、质量、能量等)以及物体内部或物体间的各种约束(位置或方位约束、动力学约束、能量约束)等。除这些用于确定物体运动和变化的特性外,也可将完善的图像绘制技术置于对模型的描述中。一般用粒子系统、连续体或刚体的受约束动力学来描述。

基于物理的造型的控制是一个重要问题。如果模型的行为是模型所固有的,那么模型将按照它自己的规律进行行为变化。在数学上,模型的行为变化常用微分方程组描述,用户除给定微分方程组的初值外,很少有其他的控制手段。如果想对模型的行为有更多的控制,必将对行为的真实性产生一定影响。如何在这两者之间有一个合理的折中,是造型的控制方法要解决的问题。目前,基于物理的造型的控制多采用基于约束的技术,允许用户规定一个约束集,要求模型的行为满足这些约束。当有多个约束时,应给出优先级,以使最重要的约束首先被满足。约束的规定是复杂的,某些约束可由一组数学等式或不等式给定,但更多的是采用微分方程组描述。

基于物理的造型是涉及**计算机图形学**、**应用数学**、**物理学**、**计算力学**等多种学科的技术。当前,它主要应用于**基于物理的动画**、**人体动画**和**科学计算可视化**等领域。

参考文献

1. Foley J D, Dam A V, et al. Computer Graphics: Principles and Practice. Second Edition. Addison Wesley, 1996

2. Barzel R. Physically Based Modeling for Computer Graphics a Structured Approach. Academic Press, 1992

(王裕国)

jìzhūn chéngxù

基准程序 (benchmark) 对计算机系统某个方面的性能进行综合测试的程序,是计算机性能评

价的工具和方法。

测试计算机整数运算处理能力的基准程序目前使用最多的是 Dhrystone 基准程序, 它的测试结果用 Dhrystone 指令每秒 (DIPS) 来表示。由 SPEC 合作组织提出的一整套基准程序以相对于 VAX 11/780 运行时间的几何平均值来测试机器的性能, 测试结果用 SPEC 比率表示 (参见 SPEC 基准程序)。

测试浮点数运算能力的基准程序目前使用最多的是 LINPACK 基准程序, 测试结果用百万 [次] 浮点运算每秒 (MFLOPS) 表示。

测试计算机系统执行事务处理能力的基准程序是 TPC 基准程序, 测试结果用事务处理每秒 (TPS) 表示。目前有专门的事务处理委员会 (TPC) 负责建立事务处理的基准程序、测试机器和公布结果等工作。

综合测试机器性能价格比的基准程序是 AIM, 测试结果被计算机制造商、公司、用户和政府部门视为参考的准则。 (詹文岛)

jiguang zhaopaiji

激光照排机 (laser beam image typesetter)

一种利用激光光束对感光材料进行扫描曝光以获得文字版面的设备。如果采用不同的控制方法, 这种设备不仅可以产生文字版面, 而且还可以产生线条图形或网点图像, 所以它具有比光机式照排机和阴极射线管照排机更为广泛的用途和发展前景。

激光照排机的主要技术指标是: 输出扫描精度、重复误差、幅面大小、输出速度等。目前大多数激光照排机的输出扫描精度均可达到每英寸一千线 (点) 以上。

激光照排机是激光照排系统的重要组成部分。激光照排系统从前端的排版系统接受排版文件之后, 在有关软件的支持下, 从精密字库中读取字模 (参见汉字字汇) 数据并发送到栅格图像处理器 (RIP), 由 RIP 对字模数据进行字形的还原、变倍和变形等一系列处理, 并按照激光扫描分辨率产生字形输出点阵。每个字符的字形输出点阵被逐个存入版面扫描缓冲存储器的指定位置。存储器中每个存储位对应版面上的一个激光扫描曝光光点。整个版面全部字符的字形点阵装入存储器后, RIP 即自动启动激光照排机, 并以同步速度逐点逐行向激光照排机发送版面点阵信息。激光照排机按新接受的版面点阵信息控制激光扫描光束, 光束的有无与版面点阵是严格对应的。扫描光束对感光软片的扫描曝

光就产生出所需要的版面。在相应的硬件和软件支持下, 通过 RIP 可以很容易地在文字版面中插入线条图形和网点图像, 实现图文合一输出。

激光照排机按其扫描方式可分为滚筒扫描式和转镜扫描式两种。滚筒扫描式激光照排机将单张软片吸附于滚筒上, 滚筒匀速自动转动时, 光学拖板在等距丝杆的拖动下匀速横移, 光学拖板发出的激光扫描光束在软片上扫描曝光形成版面。平板转镜式激光照排机的核心器件是一个高速旋转的多面锥体转镜, 激光扫描光束照射到锥体多面转镜的某一个小平面上之后又被反射出来, 经过透镜聚焦到感光软片表面, 并随着转镜的旋转而对软片扫描。每横向扫描一线, 软片也走过一线, 两者同步完成版面的扫描。

参考文献

中国印刷及设备器材工业协会. 印刷科技实用手册. 北京: 印刷工业出版社, 1992 (阳振坤)

jicheng dianlu

集成电路 (integrated circuit, IC) 通过微电子工艺将多个元器件集成在半导体芯片或陶瓷基片上, 封装在一个外壳内, 执行特定功能的电路。通常所称的集成电路至少包含一个有源元件。所谓有源元件是指能对电压、电流起控制作用并具有非线性特性的元件, 例如晶体三极管。

在集成电路中最重要的元件是用半导体材料制成的各种晶体管。

半导体有 P 型半导体和 N 型半导体两种, 把这两种半导体结合起来就可做成半导体器件。具有两个端口的半导体器件称为二极管, 它是一种无源元件, 常用的有 PN 结二极管和肖特基二极管。三端口的半导体器件称为三极管或晶体管, 它能对信号起放大或开关作用, 是一种有源器件。目前常用两种类型的晶体管: 一类是双极型晶体管, 如 PNP 晶体管、NPN 晶体管; 另一类是单极型晶体管, 又称场效应晶体管 (FET)。计算机集成电路中使用的场效应晶体管大部分是一种绝缘栅场效应晶体管, 又称金属-氧化物-半导体场效应晶体管 (MOSFET), MOSFET 又分为 PMOS 晶体管和 NMOS 晶体管。双极型晶体管工作速度高, 但结构复杂, 所以集成度低; 而 MOSFET 具有输入阻抗高、噪声小、抗辐射能力强、制造工艺简单的特点, 容易实现高集成度, 但工作速度较低。

一块 IC 可完成一定的电子线路功能, 图 1 是一

个 CMOS 反相器电路图,图 2 是其对应的 N 阱工艺的 CMOS 反相器结构图。该反相器由 PMOS 管构成的上拉网络和 NMOS 管构成的下拉网络组成。对于有效的逻辑输入,当 CMOS 反相器中的一个晶体管导通时,另一极性相反的晶体管必然截止,从而在 CMOS 电路中消除了静态功耗,并且输出电压的摆幅几乎等于电源电压。MOS 工艺最有利于大规模集成和超大规模集成。

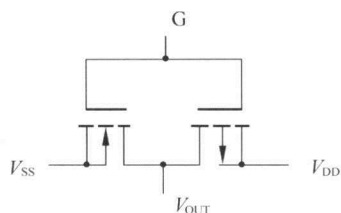


图 1 CMOS 反相器电路图

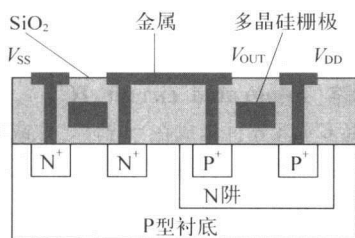


图 2 N 阱 CMOS 反相器结构图

发展历程

早在 1900 年前后,人们就发现了一类具有整流性能的半导体材料,并成功地制作出了检波器。但这些早期的晶体检波器性能不稳定,很快被淘汰了。20 世纪 30 年代,由于微波技术的发展,为了超高频

波段检波的需要,半导体材料才又引起人们重视,并制造出了锗和硅的微波二极管。1947 年,在美国的贝尔实验室由 W. Shockley 领导的一个小组发明了点接触三极管,这是世界上第一只晶体三极管,标志着电子工业从电子管时代开始进入到晶体管时代。集成电路的设计思想是英国的 G. W. A. Dummer 首先提出的。1958 年美国 TI 公司宣布制成第一块集成电路,该电路仅有 12 个元件。从此,电子工业进了 IC 时代。

集成电路技术始终是沿着集成度提高、圆片直径增大、特征尺寸减小、互连线层数增多等方向发展。迄今为止所遵循的主要规律是摩尔定律,即芯片的集成度大体上每 18 个月翻一番。实际的情况是每个芯片上的晶体管数每年增加 50%,或每 3.5 年增加 4 倍;特征尺寸(沟道长度)、门延迟、连线的步径(线宽加间距)每年减少 13%。

在硅集成电路的发展历程中,微处理器(MPU)、动态随机存取存储器(DRAM)具有代表性,20 世纪 70 年代初问世的第一代微处理器 4004,采用 10 μm PMOS 工艺,总线位数 4 位,工作电压 12 V,集成度(每块芯片集成的元器件数)为 2 300 个,时钟频率 0.108 MHz。经过 6 μm 、3 μm 、1.5 μm 、1.0 μm 、0.8 μm 、0.5 μm 、0.35 μm 、0.25 μm 、0.18 μm 等阶段,2002 年推出的 Pentium4 采用 0.13 μm ,集成度为 55×10^6 个,总线位数 64 位,时钟频率达 3000 MHz,在 31 年的时间里,特征尺寸缩小 77 倍,集成度提高 2.4 万倍,时钟频率提高 2.8 万倍。DRAM 的发展表现出同样的态势,从 1970 年的 1 Kb 发展到 21 世纪初的 512 Mb,容量提高 51.2 万倍,特征尺寸从 8 μm 发展到 0.09 μm ,缩小 89 倍,集成度从 4×10^3 发展到 524.288×10^6 ,提高 13 万倍。MPU 和 DRAM 发展进程如表 1、表 2 所示。

表 1 MPU 发展进程表

年份	产品 (型号)	线宽/ μm	晶体管数/ 10^3	单元结构		
				总线位数	时钟/MHz	电压/V
1971	4004	10	2.3	4	0.108	12
1972	8008	10	3.5	8	0.2	12
1974	8080	6.0	6.0	8	2	12
1976	8085	3.0	6.5	8	0.37	5
1978	8086	3.0	29	16	5~10	5
1979	8088	3.0	29	16/8	5~8	5
1982	80286	1.5	134	16	6~12	5
1985	80386 Dx	1.5	275	32	16~33	5

续表

年份	产品 (型号)	线宽/ μm	晶体管数/ 10^3	单元结构		
				总线位数	时钟/MHz	电压/V
1989	8048 dx	1.0	1 200	12	25 ~ 50	5
1992	80486 Dx2	0.8	1 200	32	50 ~ 66	5
1993	Pentium	0.8	3 100	32/64	60 ~ 66	5
1994	80486Dx4	0.5	1 600	32	60 ~ 66	5
1995	Pentium pro	0.35	5 500	32/64	150 ~ 200	3.3
1997	Pentium II	0.35	7 500	32/64	233 ~ 300	2.8
1998	Celeron	0.25	19 000	32/64	300 ~ 333	
1999	Pentium III	0.18	28 000	32/64	500 ~ 733	1.65
2000	Pentium IV	0.18	42 000	32/64	1 400 ~ 1 500	1.7
2001	Pentium IV	0.13	55 000	32/64	2 000 ~ 2 200	1.5
2001	Itanium	0.18	25 000	64	733 ~ 800	
2002	Pentium 4	0.13	55 000	32/64	2 000 ~ 3 000	1.5
2002	Itanium II	0.13	220 000	64	900 ~ 1 000	
2003	Pentium 4	0.09	>55 000	32/64	>3 000	1.2

表2 DRAM 发展进程表

年份	产 品 (容 量)	线宽/ μm	晶体管数/ 10 ³	芯片面积/ mm ²		
1970	1Kb	8.0	4	9.7		
1974	4Kb	8.5	8	14.5		
1976	16Kb	5.0	16	19.4		
1979	64Kb	3.0	66	31.0		
1982	256Kb	2.0	262	45.0		
		1.5		25.3		
1986	1Mb	1.2	1 049	70.0		
1988	4Mb	0.8	4 194	95.0		
1991		0.5		37.1		
1994		0.35		18.1		
1996		0.30		13.4		
1991	16Mb	0.5	16 777	130.0		
1994		0.35		63.7		
1996		0.30		46.8		
1994	64Mb	0.35	67 109	150.0		
1996		0.30		110.0		
1997		0.25		77.0		
1998		0.23		65.0		
1999		0.18		40.0		
2000		0.15		36.0		

续表

年份	产品 (容量)	线宽/ μm	晶体管数/ 10^3	芯片面积/ mm^2
1997	128Mb	0.25	131 071	102
1998		0.23		86
1999		0.18		53
2000		0.15		40
2001	256Mb	0.13	262 144	35
1998		0.23		173.0
1999		0.18		106.0
2000		0.15		73.0
2001		0.13		55.0
2002		0.11		40.0
2003		0.09		35.0
1999	512Mb	0.18	524 288	244.4
2001		0.15		174.9
2002		0.13		134.5
2003		0.11		95.0
2004		0.09		68.5

集成电路分类

根据所用晶体管的结构与工作原理的不同,集成电路可分为两大类。一类是基于双极晶体管的,通常采用晶体管-晶体管逻辑(TTL)或射极耦合逻辑

辑(ECL)形式。另一类是基于金属-氧化物-半导体(MOS)工艺的,有PMOS、NMOS以及CMOS等。此外,还有一些IC将双极电路和MOS电路做在同一芯片上,便出现了双极结构型场效应晶体管(Bi-FET)、双极金属-氧化物-半导体(BiMOS)、双极绝缘栅场效应晶体管(BiGFET)和线性CMOS(LinCMOS)等电路型式。

IC按其不同功能,可分为数字逻辑IC、模拟IC和数模混合信号IC。数字逻辑IC是基于布尔代数,以二进制记数为基础进行数字和逻辑运算的一类IC,一般由各种门电路和记忆元件组成。模拟IC所处理的信息为连续变化的物理量如电压、电流、温度等。数模混合IC既包括数字电路又包含模拟电路,因此又可分为两个系列,即混合信号IC和数字化模拟IC。

数字IC主要由门电路组成,因此可按每个芯片上集成的等效门电路个数或元器件个数表征该芯片的集成度。通常按集成度将IC分为6类(参见数字集成电路)。

IC根据其结构的不同,可分为单片IC、膜IC和混合IC。单片IC又称为半导体IC,是将所有有源、无源元件全部制作在一块半导体基片上。膜IC是用薄膜或厚膜工艺在绝缘基片上制成的电阻、电容等无源元件及其互连线组成的电路。所谓薄膜一般指厚度在 $1\mu\text{m}$ 以下的膜,厚膜指大于 $1\mu\text{m}$ 的膜。混合IC是由半导体集成工艺与成膜工艺相结合的集成电路,其工艺过程是在玻璃或陶瓷基片上制作电阻、电容等,再在同一基片上组装分立的半导体器件芯片或单片IC。混合IC在各种模拟应用中比较流行。IC的各种分类如图3所示。

与分立的晶体管电路相比,IC的优点是:①体积小,节省电路板空间;②全部电路元件位于同一基片上,温度 and 性能良好匹配;③元件间内部连线短,优化了速度和性能特性;④外连线减少,增加了电路可靠性;⑤功耗和热耗散较低。

集成电路研制

集成电路中包含有几十万、几百万乃至几亿个晶体管,从逻辑设计、电路设计、器件设计、工艺设计、版图设计到掩模制作、模拟验证等,有巨大的设计工作量和很复杂的过程。其设计和制造所需的设备虽要很大投资,但一旦开发完成并形成生产线,即可实现规模化工业生产,使生产成本降低,特别是当生产进入稳定期,成品率明显提高时,产品价格将大大降低。

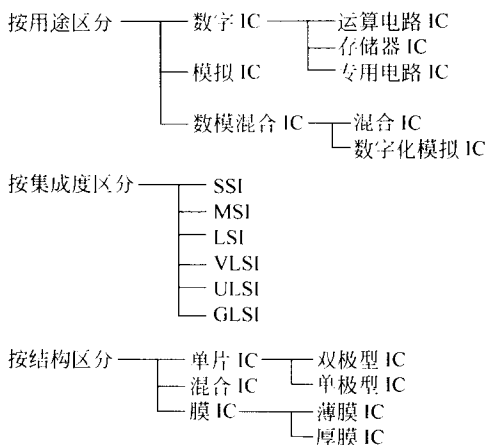


图3 IC分类

20世纪70年代到80年代,集成电路的设计理论没有很大的创新。无论是60年代至80年代的计算机辅助设计(CAD),还是70年代至90年代的计算机辅助工程(CAE),它们解决的只是将已有的电路转化成版图。而从80年代中期开始的电子设计自动化(EDA)则是从系统描述开始,按照一定的规则导出逻辑图,并在生成过程中解决可测试性(DFT)和低功耗等问题。90年代中期开始逐渐演化生成的SOC设计方法,是一种以CPU(或DSP)为核心的集成系统设计方法,强调在虚拟的原型设计环境中验证系统并实现系统集成,同时将设计和测试一体化。

集成电路生产由集成电路设计、集成电路制造和集成电路封装三个独立的工序组成。集成电路设计是将系统逻辑与性能要求等转化为具体的物理版图的过程。集成电路的制造是集成电路生产的核心,包括从晶片制备到制成(中间测试)合格电路芯片的过程(参见集成电路制造)。集成电路封装是集成电路生产的后工序,即电路芯片制成后进行装架、压焊、密封、检测和打印等。

设计和制造数量不大的专用集成电路时,价格很高。因此,可以预先制造一些具有不同规模和结构的逻辑阵列,然后根据设计要求断开一些内部连线,方便地制作出满足一定应用要求的IC,这就是专用IC(ASIC)。

随着半导体加工工艺的不断进步,集成规模越来越大,线条宽度越来越窄,已进入纳米范围。集成电路正朝着集成度更高、速度更快、体积更小、耗电量更少的方向发展,同时不断完善和提高各种特性。

多年来,硅材料的集成电路是集成电路的主流,但新材料正表现出很吸引人的优势。比如砷化镓集成电路(参见**砷化镓集成电路**)、锑化铟集成电路、砷化铟集成电路等,它们有比硅半导体电路更高的速度,特别是砷化镓半导体材料是很有希望的材料。

当物质处于低温时,电阻变为零,出现超导现象。利用超导现象可做成的超导集成电路(参见**超导集成电路**),其速度比硅集成电路快得多,且耗电极低。

参考文献

1. 王阳元主编. 集成电路工业全书. 北京: 电子工业出版社, 1993

2. 罗胜钦. 数字集成系统芯片(SOC)设计. 北京: 希望电子出版社, 2002 (时万春)

jicheng dianlu ceshi chengxu

集成电路测试程序 (integrated circuit test program)

一种以实施集成电路测试、测量和验证为主要目标的应用程序。其直接目的是使生产者和用户确定该集成电路是否具有相应的功能, 是否符合规定的性能、指标要求, 以及是否满足一定的设计规范要求; 一些测试程序还具有故障隔离的功能。一般讲, 测试程序应是一个完整的定义和指令序列。

测试语言 测试语言是一种编程用的专用测试程序设计语言, 它具有两种基本设计功能: 一种由算法和逻辑语句构成, 用于控制测试进程; 另一种由测试和测量语句构成, 用于设置并执行功能测试和参数测量。测试语言有 4 种类型: 机器语言、高级语言、命令语言和过程语言。

机器语言 机器语言可用于软件编程, 编程采用固定字和可变句两种格式。固定字格式实际上就是二进制位序列, 用以控制系统内某些功能。固定字方法是基于测试系统内大量的寄存器, 它们可由外部设置并且可控制系统的全部功能。使用可变句格式编程是基于每种功能都有相对应的相互独立的寄存器(字)。使用机器语言编程时, 这些语言(或代码)都采用操作码的形式, 都是些二进制数码, 所以无需经过翻译就能被计算机执行, 即检测、读出、解释或识别等。

高级语言 使用高级语言编程时, 机器代码仅是源语言或源代码经过汇编或编译处理后的结果格式。源语言是面向过程、面向问题的, 是一种可直接编程的公共语言。源程序被汇编或编译成一种目标程序的形式, 即以机器语言表示的形式。高级语言

是更加独立于硬件的, 对不同的计算机或测试系统都是较合理和实用的。一个有公共语言能力的系统都有自动的内部翻译器, 它可以接受公共语言并把它翻译成可执行的、自己的机器语言; 但测试系统大多不能直接使用这些翻译器, 而需重新设计。

命令语言 命令语言也是一种源语言, 该语言的主体是一些过程命令, 这些过程规定了需要执行的测试功能。比如希望 1 号程序电源产生 2.52 V 电压, 其机器代码可以表示为 00010010010100101010, 以十六进制表示则为 1252 A。若以源语言(助记符)表示, 可以写成“SPS 1, VAL”, 其中 SPS 为程序电源, 1 表示 1 号, VAL 指数值(这里, VAL = +2.52)。很明显, “SPS 1, VAL”或“SPS 1 252”的编程方式比机器语言方便、直观。上述表示方法也常用英语语言表示为“SET POWER SUPPLY NO. 1 to +2.52 VOLTS”, 或者以更精简的方式, 表示成“SET PS 1. eps, +2.52”。

过程语言 很多测试系统将命令语言和过程语言组合, 以表示一种严格定义的规则。比如, 当满足一定条件时, 程序电源 1 取其程序值的倍乘, 可表示为

IF X EQ Y THEN;

VAL = VAL · VAL;

SET PS 1, VAL;

上述表示中, “EQ”表示两个值的比较, “=”表示“VAL”值是原“VAL”值的倍乘。若在执行此语句前 VAL 为 2, 则执行后 VAL 为 4。

测试程序

测试程序准备 测试系统的测试程序都用源语言书写, 然后根据不同的软件系统结构, 将其汇编、编译或解释。汇编程序是一种符号的翻译程序, 即把汇编语言编制的源程序翻译成机器语言程序, 后者可直接在计算机上执行。编译器也是一种机器语言程序, 其功能也是将源程序转换成机器语言程序。与汇编语言的主要区别是: 汇编程序产生的机器语言与源语言是一一对应的, 而编译程序则是多对一的, 即源语言的每个语句等价于多条机器语言。编译程序可使测试系统使用更加方便、简单, 但程序本身却更复杂了。解释程序是一个处理源程序的程序, 其功能不是把输入的源程序翻译成目标程序, 而是对输入源程序中的语句一边解释一边执行, 特点是使用方便, 适于人机对话, 但占用机器时间多。

编辑器是一个常规的工具, 主要对输入数据和源程序进行编辑, 以满足测试工程师调试或修改程序的需求。

测试程序组成 集成电路测试程序由四部分组成:第一部分建立测试状态;第二部分使用很少的测试系统时间迅速地将影响测试的故障部分隔离;第三部分是测试的主体部分,包括全部测试内容,应能检查出硬失效、软失效和潜失效;第四部分为测试结果数据采集、比较、分类,分析型测试结果的数据处理,以及制图、制表、打印结果报告等。图1为一典型大规模集成电路和超大规模集成电路(LSI/VLSI)测试程序的总体布局,理想的情况是一个程序能满足全部测试要求。

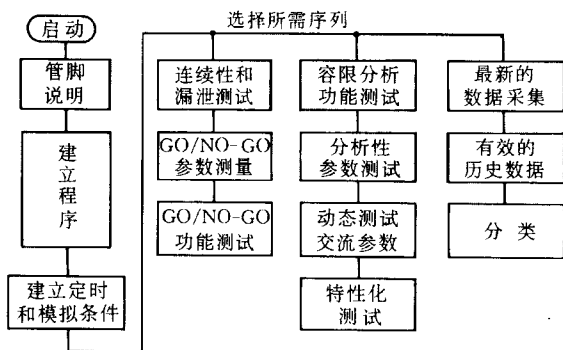


图1 测试程序总体布局

(1) 管脚说明 对测试系统管脚、被测集成电路管脚的配置进行说明,以便系统和被测集成电路之间建立相互对应关系。

(2) 建立程序 定义和设置测试程序执行时所必须的各种测试条件和资源条件,如施加、测量、比较限数据及确定各种定时功能等都是定义和设置的内容;同时还定义与实际应用有关的测试程序条件,如成品测试、中间测试、质量控制、筛选、特性测试等测试程序条件。

(3) 定时和模拟条件 规定被测集成电路的引脚连接、程序电源、定时发生器等。

(4) 连续性和漏泄测试 确保相应的引脚连接是有效的(特别是当外接自动上料器和自动探针台的情况),并及时发现由于过量漏泄而可能引起的总失效。

(5) GO/NO-GO(通过或不通过)参数测量 包括规范参数及电源功耗测量。

(6) GO/NO-GO功能测试 用尽可能短的测试序列和测试时间,尽可能完全地执行一个功能测试,以确定该电路是否满足实际应用的功能条件。

(7) 容限分析功能测试 在不同电压和定时条

件下,选择最坏图形组合下的功能测试,用以分析电路的性能。

(8) 分析性参数测试 更严格的直流参数测试,包括动态输出阻抗、功耗电流测试。

(9) 动态测试 是交流参数测试,包括上升时间,下降时间,延迟、建立、保持、存取时间,及其他反映电路时间特性的测试。

(10) 特性化测试 包括两个内容,一是确定电路的工作区,一是提供与数据表之间的交互转换。主要有基于模拟参数和定时参数的单阈值搜索和基于工作区的双阈值搜索。

(11) 最新的数据采集 范围很宽,可以是任何一次测试中的实际测量值,也可以是单参数、双参数或三维参数搜索数据,这些数据是以绝对值、图形、图表、波形、直方图、Shmoo图或报表等表示的结果形式。

(12) 有效的历史数据 包括标准偏差、平均值、峰值、参数分布等,也可以是一个或几个被测电路的一次或一组测试的三维 Shmoo 图。

(13) 分类 在测试序列结尾处,或实时地在测试执行期间分析测试结果数据并确定该被测集成电路的分类。

测试程序生成 当前,计算机辅助设计(CAD)与计算机辅助工程(CAE)已广泛用于产品发展的全过程,在测试领域广泛应用的是计算机辅助测试(CAT)。测试自动化是将被测集成电路的基准电气特性存储在计算机的存储器中,测试时与自动测试系统环境中运行被测电路的实际特性进行比较,并给出合格、失效或其他判断。为此,需要借助计算技术建立被测集成电路基准电气模型;需要计算机对自动测试系统进行测试数据控制和测试过程控制;需要用计算机将测试结果数据进行相应计算和分析,以给出生产质量控制数据、生产过程控制数据,并进行生产管理。上述过程分别对应 CAT 的三个主要构成系统,即测试设计系统、测试执行系统(自动测试系统)和测试数据管理系统。测试和设计链接就是这个集成 CAT 系统的基础环节。一个产品开发的过程,可以分为设计、工程、测试、生产等环节,每个环节使用或产生的数据往往是另一环节所需要的。比如 CAD 数据包含有大量关于芯片的物理信息,这些信息对该芯片的测试、诊断和工艺修复就是很需要的;又如芯片逻辑模拟或故障模拟时生成的设计验证文件或测试生成文件就可作为测试程序的基础。

测试设计系统的核心是 CAD 系统中的模拟

器,它有逻辑模拟、故障模拟、器件模拟或工艺模拟等多种。测试执行系统的主体是运行测试程序进行测试的自动测试系统。由于历史发展的原因,当今各种模拟器的开发和运作都是分别独立进行的,而测试系统作为一种测量、测试仪器,其测试的原理、功能、指标、规格说明以及组成、结构等表现出更大随意性。在当前的工程应用中,绝大部分属于多模拟器多测试系统环境,因此不仅需要特定的模拟器到特定型号测试系统(系列)测试程序的生成,还要考虑多模拟器多测试系统(系列)之间交叉测试程序生成,以及多测试系统(系列)之间测试程序的交互移植。由 n 个模拟器和 m 个测试系统(系列)组成的多通路链接,需要相应 $n \times m$ 个格式转换器。如果每当一个新的格式进入 CAD/CAE 或者 CAT 环境时,必须立即补充开发一个新的转换器集合,其结果将是开发和维护程序的成本大幅度提高,而且这个庞大和相互独立转换器集合的管理将非常困难。为了解决测试和设计之间这个多通路链接的 $n \times m$ 复杂性问题,合理的办法是设计和实现一种中间(中介)格式。原来需要从每一种模拟器文件转换到每一种测试系统(系列)测试程序,现在只需要转换到中间格式;当需要目标测试系统测试程序时,只需将这个已经没有不同模拟器属性的中间格式程序进行划一的测试程序转换即可。多通路链接的转换器复杂性降为 $n + m$ 。图 2 是具有中间格式的双向链接模型。链接必须是双向的,以便测试结果数据返回到设计部门,便于设计工程师对原始设计数据进行必要的修改,以进行再模拟。复杂性的降低可节省相当的资源和相应的维护开销。

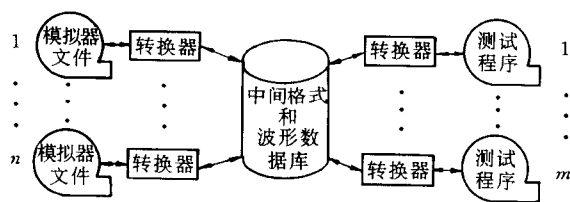


图 2 双向测试和设计链接模型

近 10 年来,由于 EDIF, CADDIF, WAVES 等不同类型电气数据交换标准格式或中间格式的推广,测试和设计链接新结构取得了长足进展,很多在市场上流通的产品都具有相应的链接能力,在减少产品从设计到上市时间,提高产品测试质量等方面,取得了明显的效果。

参考文献

Healy J T. Automatic Testing and Evaluation of Digital Integrated Circuits. Virginia: Reston Publishing Company, 1981 (时万春)

jicheng dianlu ceshi xitong

集成电路测试系统 (integrated circuit test system)

测试集成电路(IC)直流参数、交流参数和逻辑功能的一种设备。现代集成电路测试系统集成计算机技术、自动化技术、通信技术和精密电子测量技术于一身,形成了技术密集型的自动测试设备(ATE)产业。IC 测试系统的基本功能是检测被测器件(DUT)的工作特性是否符合器件生产厂家给定的特性要求。系统按其测试的对象不同,大体分为数字 IC 测试系统、模拟 IC 测试系统和数模混合信号 IC 测试系统。

数字 IC 测试系统 根据应用领域的不同,可分为通用 IC 测试系统、IC 验证系统和验证测试系统。

对通用 IC 测试系统的要求是测试的通用性、技术指标的综合性。IC 验证系统是一种对少量试制样品进行功能验证的测试设备,主要用于专用集成电路(ASIC)设计。验证测试系统是在 IC 验证系统的基础上,增加了直流和交流参数的测试功能,其特点是将 IC 的设计与测试紧密地联系起来。

测试系统由硬件和软件组成。

系统硬件 可分为测试计算机子系统和测试、测量子系统两大部分,其结构如图 1 所示。

测试计算机子系统包括以下部件:

(1) 系统控制器 一个高速处理部件。通过总线结构对测试系统进行控制,可以完成各种测试、记录和数据处理任务。系统控制器可以采用各种商用计算机。

(2) 主存储器 与系统控制器相连的高速大容量存储器,用于存储各种原始的测试向量和数据。

(3) 图形发生器 控制所有测试图形的产生和控制图形序列的处理器,提供测试功能数据,功能指施加、比较、禁止、屏蔽等。

(4) 时钟发生器 一个具有多相时钟信号的多重定时集合发生器,定时集合可按测试周期实时选择和切换。

(5) 外围设备 用来满足控制器测试环境要求的部件,如磁盘、磁带、打印机、图形终端等。

(6) 矩阵 模拟开关矩阵,通过 $50\ \Omega$ 矩阵接于测试台。

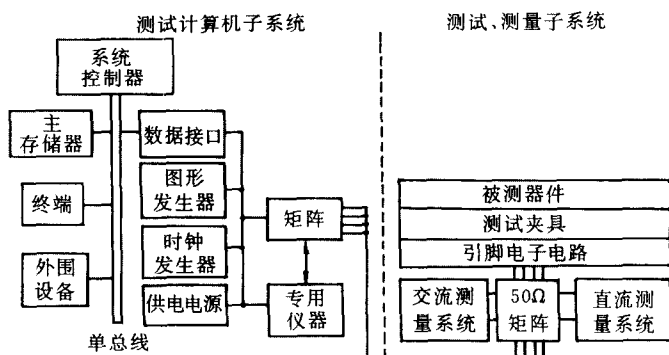


图1 数字IC测试系统框图

(7) 数据接口 主控器与测试台之间的接口,用于测试数据、测试命令等信息交换。

(8) 供电电源 可编程供电电源的集合。

测试、测量子系统包括以下部件:

(1) 引脚电子电路 由输入驱动器、输出比较器和动态负载等组成。

(2) 测试夹具 由配线插座、测试模板组成,能将DUT的引脚连接至相应通道的驱动器、比较器、电源或地。

(3) 50 Ω 矩阵 多通道转换矩阵,能为测试系统的所有引脚提供双向选择。

(4) 交流测量系统 在系统控制器的指挥下,控制引脚电子电路完成定时测量的逻辑部件。

(5) 直流测量系统 在系统控制器的指挥下,控制引脚电子电路完成直流参数测量的逻辑部件。

系统软件 按功能可分为以下5种:

(1) 操作系统软件 测试系统核心部分之一,由测试系统驱动程序和管理程序构成。前者直接控制测试,后者用于监控操作员的请求,控制输入输出设备的使用以及编辑测试程序等。

(2) 系统的实用软件 包括数据的采集、管理和处理程序。可以通过Shmoo图表示不同参数(如电压与定时)之间的关系,通过直方图显示DUT的通过与不通过(PASS/FALL)统计结果,还可显示测试向量。

(3) 自诊断和校准软件 为测试工程师提供足够的测试、诊断和校准的工具。该软件可以验证测试系统本身硬件的操作特性、精度、容限和极限,并可通过自校准保持系统的高精度。

(4) 测试程序 由测试主程序和功能测试图形构成。不同的测试系统,其编程语言和方法不尽相同。

(5) 测试程序调试与开发软件 加速测试程序开发的软件工具,比如计算机辅助设计逻辑模拟输出与测试系统之间的数据格式转换软件,不同测试系统间的测试程序移植软件以及脱机测试程序开发软件等。

模拟IC测试系统 基本结构(图2)由控制器、源单元、测量单元和测试头组成。

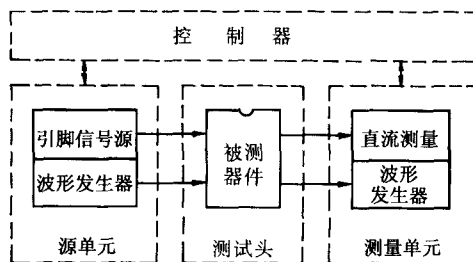


图2 模拟IC测试系统框图

(1) 控制器 模拟IC测试系统的核心部件。通过总线对测试系统进行控制。

(2) 源单元 向DUT提供直流信号和各种激励波形。

(3) 测量单元 测试DUT对各种激励信号的响应,包括直流测量、波形测量和时间测量。

(4) 测试头 DUT的系列测试盒。

模拟IC测试系统输出响应部分所需的测量仪器品种繁杂,这给测试带来一定的困难。随着计算技术的发展,模拟IC测试系统已广泛地采用了数字信号处理(DSP)技术,由计算机软件取代了系统内的一些信号源和测量仪器的功能,并由此带来测试时间短、精度高等优点。

数模混合信号IC测试系统 系统结构(图3)与模拟IC测试系统基本相同,也是由控制器、源单

元、测量单元和测试头组成。与模拟 IC 测试系统不同的是源单元中配置了数字图形发生器,测量单元中加入了数字比较器,以供数模混合信号 IC 中数字电路部分的测试。数模混合信号 IC 测试系统的功能和特点说明如下:①控制器与各子系统相连,提供所需的测试程序。②由于采用了 DSP 技术,系统具有波形合成器和波形数字化器的功能,此外,还能向 DUT 的数字输入端提供数字激励,接收 DUT 输出端的输出响应,对信号进行锁相同步以使系统内

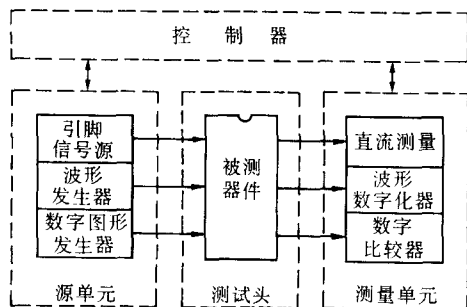


图3 数模混合信号 IC 测试系统框图

模拟信号和数字信号同步。③数据传输通过输入输出端口,激励图形和响应图形都以总线速度并行传输。④波形发生器与波形数字化器中具有多种不同的模块,它们能以不同的时钟频率运行。⑤数字部分的操作与通用的数字 IC 测试系统相类似。⑥测试头是系统的关键部分。引脚信号源、数字图形发生器和波形发生器经过测试头向 DUT 提供数字与模拟输入信号,DUT 的数字与模拟响应信号同样通过测试头反馈给直流测量、数字比较器和波形数字化器处理。

数模混合信号 IC 测试系统的测试头内,数字与模拟信号靠得很近,必须采取措施降低噪声。另外,模拟信号的测量精度要求高,信号源与被测器件之间的距离应尽量短。

随着微电子技术的发展,对 ATE 产业的要求也愈来愈高,新一代的超大规模集成电路(VLSI)通用测试系统、高速大容量存储器测试系统、数模混合信号 IC 测试系统等都已相继问世。在 ASIC 验证系统基础上发展起来的新一代 ASIC 验证测试系统由于采用最新技术和方法,其性能和指标已接近大型通用测试系统。新一代系统具有下述特点:①模块化的引脚信号结构使配置灵活;②输入输出通道分割方式可以有效利用资源;③具有扫描器件的测试能力;④VXI 总线接口可与多种外设连接;⑤能够接

收任何电子设计自动化仿真器的输出,可以快速开发测试程序。

模拟 IC 测试系统的生产呈下降的趋势,而数模混合信号 IC 测试系统发展很快。在硬件方面,正朝着模块化结构方向发展。在模拟信号与数字信号共存的情况下,降低噪声是确保模拟波形保真度的关键,严格定时是保证数字与模拟电路测试过程中保持同步的必要条件,实现自动校准是保证测量精度的必由之路。在软件方面,将进一步采用和开发数字信号处理技术。在测试程序的开发方面,已开始采用面向目标的编程技术、计算机辅助设计(CAD)与计算机辅助测试(CAT)之间的数据格式转换技术等。系统的技术指标将在进一步提高测试通道数、测试速率、定时分辨率及高电压、大电流等方面得到发展。

参考文献

1. Needham W M. Designer's Guide to Testable ASIC Devices. New York: Van Nostrand Reinhold, 1991
2. Massara R E. Design & Test Technique for VLSI & WSI Circuits. London: Peter Peregrinus Ltd., 1989

(刘鸿琴)

jicheng dianlu zhizao

集成电路制造(integrated circuit manufacturing) 按照电路功能的设计要求,采用特殊的数据微缩制版、薄膜淀积和套版蚀刻技术,将微电子设备中各种规格的元件(导线、电阻、电容和电感)和各种性能的器件(如二极管和晶体三极管)直接制造集成在一块半导体芯片上,组成一个不可分割的电路整体,完成其预定的电路功能。所谓集成是指在制造元器件的过程中同时完成对各元器件的连接,在连接元器件的过程中又同时整体地完成元器件的制造。

电子设备小型化的发展结果促使人们打破了传统的电路设计和制造概念。以往的电路设计都是将分立的元器件按要求组合和装配在一起,然后用导线加以焊接而成。1958 年美国人 J. Kilby 发明了集成电路,次年美国的得克萨斯仪器公司和仙童半导体公司分别实现了第一块集成电路。此后,集成电路经历了 20 世纪 60 年代初期的小规模(SSI)、60 年代中期的中规模(MSI)、70 年代初期的大规模(LSI)、70 年代中期的超大规模(VLSI)以及 90 年代的特大规模(ULSI)阶段。高密度、低功耗、高可靠性和低价格是集成电路的特点。在集成电路整个发

展过程中,集成度(参见集成电路)平均每18个月翻一番,器件的几何尺寸每年下降13%(称为摩尔定律)。ULSI芯片上的元器件数量已超过1亿个。集成电路内部的最细线条小于 $0.1\mu\text{m}$ 。在一块面积约 1cm^2 的芯片上集成如此多的元器件,若不借助高倍显微镜,用肉眼是难于观察清楚的。

集成电路制造的组织体系 现代集成电路制造的基础是硅平面扩散工艺,即用二氧化硅作为杂质扩散阻挡层(见图3(f))的方法来制造晶体管。采用硅单晶作为集成电路的基础材料的原因是它具有合适的禁带宽度(1.12eV)、稳定的氧化物以及在自然界中丰富的储藏量。除硅之外还采用锗、锗硅以及砷化镓、磷化铟等Ⅲ-V族化合物半导体基础材料。

现代集成电路的制造工艺一般都应在超净化工作室内进行。室内的空气保持在良好控制的温度和湿度下,并且能连续地循环和过滤。按空气中微粒净化级别的定义,在“0.5微米的M2级”的净化环境是指:空气中大小在大于、等于0.5微米的颗粒每立方米不超过100个。在关键的光刻工艺区域,净化条件应保持在“0.3微米的M1级”以下,即空气中大于、等于0.3微米的颗粒每立方米不超过3.09个。

集成电路制造需要大量的纯水以供硅片在工序前的清洗。普通水中的颗粒(矿物质、动、植物遗体等)以及钠、钙、镁、铜和铁等离子都会沉积在硅片上,导致器件性能的退化和失效。因此,采用超纯度、电子级水平的去离子水和化学试剂是制造集成电路的基本条件之一。在室温下,制造集成电路所使用的去离子水的电阻率应高于或等于 $18 \times 10^6 \Omega \cdot \text{cm}$ 。

集成电路的制造一般需要经过电路设计,工艺模拟,版图设计,衬底单晶硅片的选用,工艺集成,中间监测,划片、装架、封装,老化筛选,成品率统计和可靠性

鉴定等工序。制造集成电路的组织体系如图1所示。

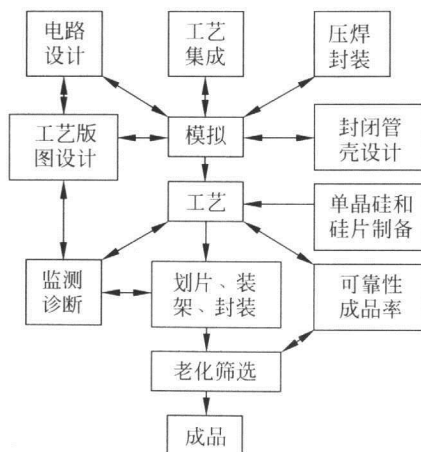


图1 制造集成电路的组织体系

集成注入逻辑(I²L)器件因其超高速逻辑运算能力,N沟道金属-氧化物-半导体(NMOS)器件因其高密度、低功耗特性而被广泛用于计算机工业中。在各种存储器芯片中,以随机存取存储器(RAM)芯片的元器件密度最高。这是因为在RAM芯片中,矩阵中任何一位的信息都能独立存取。每一列存储位可利用扩散区、掺杂多晶硅线条或金属薄膜层的导电线条来实现存取。类似矩阵的列、行则可用图中位线来存取。只读存储器(ROM)的数据是可以永久保存的,旧的信息不能清除,新的信息不能输入。静态RAM能长久地保留数据,除非电路的电源突然中断。动态RAM为保持存储的信息,要求存储在每个存储单元中的电荷数据周期性地“再生”。图2

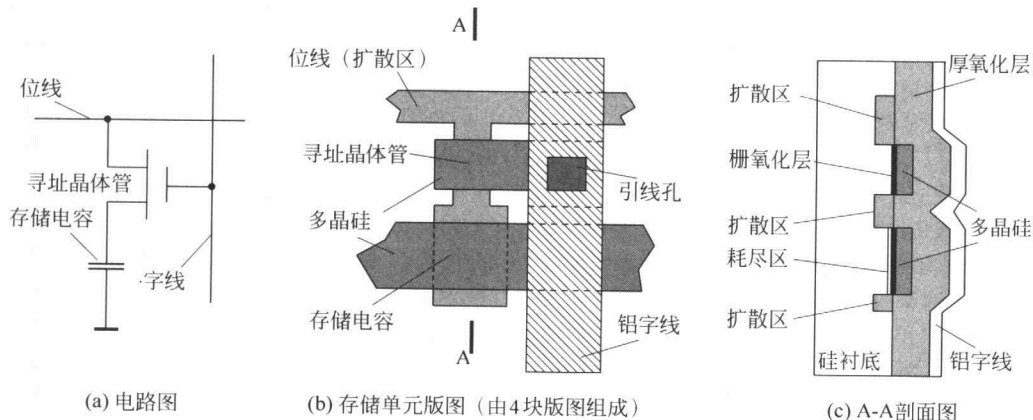


图2 单管动态存储单元

所示的是一种基本的单管动态存储单元。其中图 2(a) 为电路图, 图 2(b) 为由 4 块光刻掩模版(扩散区版、多晶硅版、孔版和铝引线版)组成的单元版图, 图 2(c) 为图 2(b) 的 A—A 剖面。由扩散区(源和漏区)形成位线, 扩散区即为晶体管的源电极。通过字线将该寻址晶体管的栅电极选通, 存储信号(1 或 0)通过位线被读写。

现代集成电路制造是个极复杂、极精密的过程。尽管一套给定的流程包含的工艺步骤可以多达百余道, 但是, 只要工艺控制严格, 重复性好, 成品率可以较高; 加上每一批流程中有近百片的硅片, 每一片硅片上又有几百、几千粒 VLSI 的管芯(产品)同时在一起加工, 因此每个产品的成本还是相当低的。

集成电路制造工艺 在集成电路制造中, 电路设计、工艺模拟和版图设计是制造前的准备阶段。它们不仅关系到最终完成的集成电路能否按设计要求完成指定的功能, 还关系到产品的成品率的高低。集成电路制造工艺可以粗略地分成以下几个重要过程。

单晶硅和硅片的制备 单晶硅和硅片的制备是集成电路制造前期工序之一。首先用化学方法把自然界的砂子(SiO_2)制成多晶结构的电子级硅棒, 再根据晶体生长所涉及到的从固体、液相或气相到结晶固相的相变这一物理过程, 将多晶硅棒拉制成单晶硅棒, 最后经切割、磨边、双面研磨、腐蚀、抛光等工序将单晶硅棒制成单晶硅片。单晶硅片具有一定的晶向和导电类型, 而电阻率、缺陷和位错密度、含氧量等参数都需要满足一定的设计要求。

工艺集成 工艺集成是集成电路制造的核心。图 3 为集成电路制造的部分基本集成工艺。其中图 3(a) 为衬底单晶硅片, 图 3(b) 表示在硅片上用高温热生长法生长一层氧化硅(SiO_2)后再涂敷上一层抗蚀剂, 例如正性胶。图 3(c)、(d) 和 (e) 为光刻工序, 目的是将设计的掩模版上的图形精确无误地转移到 SiO_2 层上。详细过程如下: 利用照相曝光技术, 把掩模版上图形复制到抗蚀剂上, 经显影后, 掩模版上的图样空白部分所对应的抗蚀剂(也称光刻胶)被溶解掉(曝光后可溶解的光刻胶称为正胶, 曝光后不可溶解的光刻胶称为负胶)。留下的光刻胶经过烘烤、坚膜以后便具有抗蚀性能。经过等离子蚀刻把裸露的 SiO_2 层蚀刻掉, 在抗蚀剂下面的 SiO_2 被保护而留下。再除去 SiO_2 上的抗蚀剂, 最终使掩模版上的图形完整无缺地、精确地复制到氧化硅层上, 如图 3(e) 所示。图 3(f) 为掺杂剂通过高温

热扩散或离子注入的方法进入硅表层的过程。掺杂剂可以是硼、磷、砷等离子或原子。氧化硅层起着掩蔽掺杂的作用, 若采用离子注入, 氧化硅层上的抗蚀剂常被保留, 它同样具有阻挡掺杂离子或原子进入硅中的作用。被掺杂的区域叫掺杂区或扩散区。掺杂区在双极型器件中常用来形成基极、发射极和电阻; 在 MOS 器件中则常用来形成源区、漏区、N 阱和 P 阱, 还用作对夹断电压的大小控制等。图 3(g) 表示蚀刻掉 SiO_2 层, 留下所需要的扩散区域。在实际集成电路制造过程中, 常常多次执行图 3(b) 到图 3(g) 的工序, 只是用不同的版图以完成多块掩模版的嵌套。图 3(h) 为在衬底硅片上淀积上一层起绝缘和隔离作用的厚氧化层。图 3(i) 为刻孔工序, 在需要连接的各掺杂区上方将厚氧化层开个孔, 形成带引线孔的氧化层。图 3(j) 为金属化过程, 即在整个衬底硅表面溅射(或蒸发)上一层金属导电膜, 金属膜通常为铝层。图 3(k) 表示按照实际线路的要求, 光刻、刻蚀铝层, 使硅片浅表层的各元器件按设计要求而相互连接。图 3(l) 为淀积钝化层。钝化层起着与外界的空气、水汽隔绝, 保护管芯免受侵蚀和碰伤、划伤的作用。图 3(m) 为蚀刻压焊孔。按光刻、蚀刻的相同方法, 在钝化层上蚀刻出较大的压焊孔, 以便管芯在划片、装架后通过热压焊式超声压焊方法, 用直径 $20 \sim 30 \mu\text{m}$ 的铝丝将芯片各压焊点与对应的管座引线端连接。经过由图 3(a) 到图 3(m) 最基本的工艺集成, 一个集成电路的芯片被制造出来。

管芯装配和老化筛选 管芯(也叫芯片)与管座的装配、老化筛选等都是集成电路制造的后工序。装配包括划片、装架、封装等工序, 装架又分芯片的嵌装、压焊等工序, 封装包括注入填料、焊接或粘接盖板等工序。封装后再进行老化筛选, 对一个个集成电路成品进行“磨合”, 完成严格的性能检测和分类。只要在一一定的规格和等级允许的条件下使用, 就可确保每一个集成电路产品都有较高的稳定性和可靠性。

在集成电路制造的各个工序环节, 几乎都有相应的工艺质量监测和分析诊断技术。严格的质量管理和控制措施是保证集成电路产品的高质量、高可靠性以及高成品率的必要条件。

集成电路制造的关键因素 集成电路的迅速发展在很大程度上决定于设备和技术的发展。一代的设备, 有一代的工艺; 一代的技术, 有一代的产品。与日俱进的超精细加工、薄膜淀积和离子注入的设备和技是集成电路的迅速发展的三大关键因素。

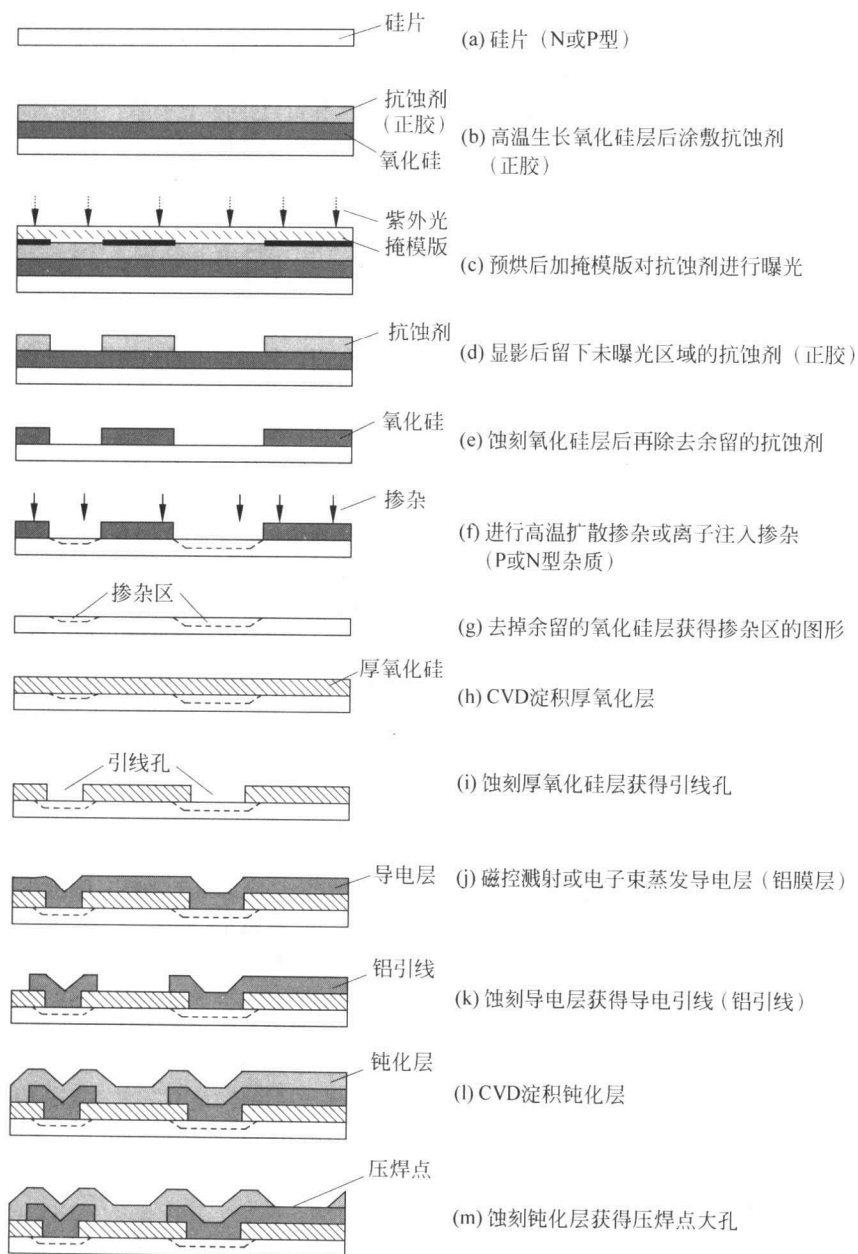


图3 集成电路制造的部分典型集成工艺

超精细加工 在超精细加工中,第一步是将精细的图形从掩模版转移到硅表面上,图形转移的质量取决于光刻机和抗蚀剂的性能及其使用方法。由接触式光刻机、接近式光刻机、扫描投影光刻机、分步重复光刻机到步进扫描光刻机,已历经了五代光刻

机的发展,由精细到微米级的紫外光光学光刻技术已发展成精细到微纳米级的电子束光刻和X射线光刻技术。超精细加工的第二步是对精细图形的蚀刻。由于湿法蚀刻技术的均匀性和可靠性较差,因此在超精细加工中逐渐被淘汰。干法蚀刻发展甚

快,干法蚀刻有等离子蚀刻(IE)、反应离子蚀刻(RIE)、离子铣(IBE)和感应耦合等离子(ICP)蚀刻等形式,其中最常用的是IE和RIE两种。这些形式的主要差别在于蚀刻的方向性、选择性及蚀刻速率。

薄膜淀积 集成电路中的薄膜淀积方法常用真空蒸发、磁控溅射、化学气相淀积(CVD)和电镀淀积薄膜。利用光照或等离子体的作用可把CVD的淀积温度降下来。由于低压化学气相淀积(LPCVD)具有稳定可靠、重复性、均匀性好,适合大批量生产等优点,因此被广泛使用。在真空蒸发基础上发展起来的分子束外延(MBE)以及在CVD基础上发展起来的金属有机物化学气相淀积(MOCVD)都用来生长极薄的外延层。

离子注入技术 离子注入是一种向硅衬底中引入可控制数量和可控制深度的杂质,以改变其电学性能的方法。离子注入能够均匀、重复控制注入杂质的浓度和深度,而且可在低于125℃下进行操作,因而在所有应用中都优于高温扩散,已成为超大规模集成电路制造中最先进的技术。不同的工艺对离子注入的要求也各不相同。在MOS器件中利用离子注入机可完成深埋层、倒掺杂阱、穿通阻挡层、阈值电压调整、轻掺杂漏区(LDD)、源漏注入、多晶硅栅、沟槽电容器、超浅结、绝缘体上硅(SOI)等各种离子注入。离子注入后需要经过退火把注入的杂质元素激活并消除因注入而引起的晶格损伤。为防止注入杂质在退火过程中向衬底内部再扩散,已越来越多地采用快速退火技术(RTA)。

展望 集成电路所用的主要衬底材料是单晶硅,但单晶硅化镓材料在集成电路中所占地位将越来越重要。各种超晶格材料不仅会改善某些现有的集成电路器件的性能,而且将产生一些有新效应的新器件。

新结构 新结构的集成电路发展趋势主要有以下几方面:①在绝缘衬底上生长单晶硅薄膜(SOI)技术。此技术最诱人的用途是制造三维集成电路,由二维发展到三维可以大大提高空间集成度。②多层布线技术。多层布线可以解决单层布线中常遇到的交叉线困境,实现多层布线的关键是解决中间绝缘层的表面平坦化问题。③耐熔金属硅化物和浅结构技术。耐熔金属硅化物与通常的掺杂多晶硅相比具有较低的自由电阻,此外,它还具有与硅接触点的长期稳定和可靠性,不易发生“穿刺”现象。④深槽结构技术。在VLSI和ULSI结构中,深槽的用途有两个方面:一是用作隔离,二是用来制作垂直方向

的电容器,使电容器所占面积进一步缩小,从而增加器件的集成度。⑤量子阱、异质结等结构将使半导体光电集成电路的性能不断地提高。

新工艺、新技术 ①为减少互连线的电阻率,降低功耗,增加运算速度,铜质互连线必将取代传统的铝质互连线。②为制造多层结构的3D高密度集成电路,必须使用先进的化学机械抛光(CMP)平坦化技术代替传统的硼磷硅玻璃回流和旋涂膜层的平坦化方法。③倒装芯片技术是将芯片的有源面(具有表面键合压点)面向基座的粘贴封装技术。它是从芯片器件到基座之间最短路径的一种封装设计,为高速信号提供了最良好的电接触。由于不使用引线框架或塑料管壳,因此重量和外形尺寸也有所减小。④为获得更低成本、更加可靠、更加快捷、更高密度的电路,必须采用先进的集成电路封装技术,它们是球栅阵列(BGA)、板上芯片(COB)、表面安装技术(SMT)、多芯片模块(MCM)、芯片尺寸封装(CSP)及圆片级封装(WLP)等。

参考文献

1. Gise P E, Blanchard R. Semiconductor and Integrated Circuit Fabrication Techniques. Virginia: Fairchild Corporation, 1979
2. Michael Quirk, Julian Serda. Semiconductor Manufacturing Technology. New York: Prentice Hall, 2001

(汪锁发)

jicheng shuzi huan zaibo

集成数字环载波(integrated digital loop carrier, IDLC) 当今公共电话网最广泛使用的将数字技术引到用户设备的**宽带接入体系结构**。IDLC利用光纤-数字技术以支持以下各种服务:速率为DS1、部分DS1、64 kb/s以及基本费率接口(BRI)的接入网。传输网的速率通常是DS1、DS3、OC-3或OC-12(参见**时分多路复用**)。开发最多的设备是SONET(参见**同步光纤网**)兼容的或基于贝尔通信研究所(Bellcore)规范的。它为现今的模拟传输过渡到基于SONET传输提供了一种途径,可同时支持基本的和增强的电话服务。各个厂家提供的硬件、软件和结构特点是不同的。一般来说,大部分设备提供商支持以下系统特点:远程软件供应能力;方便的设计以支持外部的场地运行、管理、维护和供应(QAM&P)的特点;灵活的硬件结构以适应今后软件升级;为最终用户提供数字服务。体系结构要考虑能补充**数字用户专用线(DSL)**技术,以便在

下一阶段过渡到端到端宽带网络设计,实现服务汇聚的处理。

参考文献

1. 胡道元. 智能建筑计算机网络工程. 北京: 清华大学出版社, 2002

2. Padmanand warrier, Balaji Kumar, XDSL Architecture. McGraw-Hill Inc., 2000 (胡道元)

jihe

集合 (set) 由一定范围的、确定的且彼此可以区分的对象(抽象的或具体的)组成的整体。它是数学中最重要的基本概念之一。组成集合的对象称为它的元素。例如“中国人的集合”、“长江里的鱼的集合”及“方程 $x^3 - 4x + 3 = 0$ 的实根的集合”等,都是集合的例子。

属于关系 若对象 a 是集合 A 的元素,则称 a 属于 A 或 A 含 a , 记为 $a \in A$; 否则称 a 不属于 A 或 A 不含 a , 记为 $a \notin A$ 。不含任何元素的集合称为**空集**, 用 \emptyset 表示。

子集 若集合 A 的元素都是集合 B 的元素,则称 A 是 B 的子集或 B 包含 A , 记为 $A \subseteq B$ 。

相等 若集合 A, B 的元素完全相同,则称 A 与 B 相等, 记为 $A = B$; 否则称 A 与 B 不相等, 记为 $A \neq B$, 显然, $A = B$ 当且仅当 $A \subseteq B$ 且 $B \subseteq A$ 。

若 $A \subseteq B$ 且 $A \neq B$, 则称 A 为 B 的**真子集**, 记为 $A \subset B$ 。

对任意元素 a 和集合 A , 恒有 $a \notin \emptyset, \emptyset \subseteq A$ 且 $A \subseteq A$ 。

集合的常用表示方法有以下三种:

(1) 列举法 依照任意次序不重复地列举出集合的全部元素,并用花括号括起来。例如

10 以内全体素数的集合 = $\{2, 3, 5, 7\}$ 。

方程 $x^4 - 4x + 3 = 0$ 的全部实根的集合 = $\{-1, 1, 2\}$ 。

(2) 命题法 若集合 A 的全体元素恰为使命题 $P(x)$ 真的全体 x , 则 A 可用 $\{x | P(x)\}$ 表示, 即 $A = \{x | P(x)\}$ 。例如

有理数集合 $Q = \left\{ \frac{m}{n} \mid m \text{ 和 } n \text{ 为整数且 } n \neq 0 \right\}$ 。

$\{1, 2\} = \{x | x \text{ 为方程 } x^2 - 3x + 2 = 0 \text{ 的实根}\}$ 。

(3) 归纳定义法 用这种方法定义一个集合 A 时,一般包括以下三个步骤:

① 基本项:已知非空集合 $S_0 \subseteq A$ 。② 归纳项:给出一组规则,从 A 的任意元素(一个或多个)出

发,依据这组规则所得之元素都是 A 的元素。③ A 中每个元素都可以通过有限次应用①和②得到。

步骤① 是归纳定义法的基础,且保证 $A \neq \emptyset$; 步骤② 是归纳定义法的关键步骤,用以从 A 的已知元素获得其新元素;步骤③保证所定义的集合 A “最小”,因而是惟一的。在用归纳定义法给出集合时,步骤③常常省略不写。

例如,全体奇数的集合 O 可用归纳定义法给出如: ① $1 \in O$; ② 若 $n \in O$, 则 $n + 2 \in O$; ③ 省略。

(王兵山)

jihelun

集合论 (set theory) 以一般集合为研究对象的数学的基本分支之一。集合论在数学中占有独特的地位,它的基本概念已渗透到数学的所有领域。按现代数学的观点,数学各分支的研究对象,或者是带有某种结构的集合(如群、环、域、拓扑空间等),或者是可用集合直接定义(如自然数、有理数、实数、函数等),或者是可借助集合定义(如范畴、函子、自然变换等)。因此,从某种意义上说,集合论是整个现代数学的基础。

集合论是 G. Cantor 于 19 世纪末创立的,至今经历两个阶段:1908 年以前称朴素集合论,1908 年以后又产生了公理集合论。公理集合论不外乎是朴素集合论的严格处理,由于广泛使用数理逻辑的工具,它又逐渐成为数理逻辑的一个重要分支。自 20 世纪 60 年代以来,它又获得迅速发展。

朴素集合论 它是 G. Cantor 最早建立起来的集合论,故又称康托尔集合论。所谓一个集合,按 G. Cantor 的定义,就是一定范围的、确定的且彼此可区分的对象(抽象的或具体的)汇集在一起组成的一个整体。组成集合的对象称为它的元素或成员。如果对象 x 是集合 A 的元素,则记为 $x \in A$, 否则记为 $x \notin A$ 。因为在 G. Cantor 的集合定义中,对于对象的所属范围、性质及其如何汇集都没做要求,所以它还构不成集合的严格数学定义,而是集合的一种直观描述或说明。

在朴素集合论里,无条件地接纳了以下三条基本原理:

(1) 外延原理 两个集合相等是指它们的元素完全相同;

(2) 概括原理 对任意性质 P , 都有一个使 $P(x)$ 真的全体 x 的集合, 记为 $\{x | P(x)\}$;

(3) 选择公理 对每个由非空集合组成的集合

族 \mathcal{S} ,都有一个函数 f (称为 \mathcal{S} 的选择函数),使得对任意 $A \in \mathcal{S}$ 皆有 $f(A) \in A$ 。

选择公理还有其他多种表达形式。这条公理的直观意思是说,可从任何非空集合内取得一个元素。选择公理对整个数学的影响是巨大的,它在数学的许多分支,如分析、拓扑、代数等中都是不可少的工具。

有了集合概念,就可以定义一个集合 A 的子集 $S \subseteq A$,幂集 $\mathcal{P}(A)$ 和补集 $\sim A$,两个集合 A 与 B 的并集 $A \cup B$,交集 $A \cap B$,差集 $A - B$ 和笛卡儿积 $A \times B$ 等,并可进而定义集合上的关系、集合到集合的函数及集合的基数(也称势)和序数等一系列概念。关于它们的运算和性质的研究,就构成了朴素集合论的主要内容。

集合论悖论 悖论就是逻辑矛盾的论述。在朴素集合论内,依据概括原理,对任意性质 P ,都有一个使 $P(x)$ 真的全体 x 的集合 $\{x | P(x)\}$ 。如果取 $P(x)$ 为“ x 是集合且 $x \notin x$ ”,便可获得一个集合 $S = \{x | x \text{是集合且 } x \notin x\}$ 。现在要问: S 是不是它自己的元素呢?若 S 是它自己的元素即 $S \in S$,则由 S 的定义及 S 是集合得 $S \notin S$;若 S 不是它自己的元素,则由 S 的定义及 S 是集合得 $S \in S$,总之恒有 $S \in S$ 当且仅当 $S \notin S$ 。这显然是一个矛盾,它就是著名的罗素悖论。此外,人们在朴素集合论内还相继发现了许多悖论。这类悖论表明,朴素集合论的理论是不协调的,这也使人们对整个数学推理的正确性与结论的真理性产生怀疑,酿成了数学史上的第三次危机,影响是深远的。

公理集合论 为了避免悖论,人们感到再不能把任何逻辑上可定义的概念的外延都当作一个集合了,即须对 G. Cantor 的集合定义加以限制。但一直未能找到一个简单而无问题的替代定义,于是就从当时已有的集合论成果出发,来反求足以建立这一数学分支的原则,这就是集合论的公理化研究。E. Zermelo 于 1908 年提出了第一个集合论公理系统,后经 A. A. Fraenkel 和 A. T. Skolem 加以改进和补充,形成了著名的 ZF 公理系统。B. 罗素也于 1908 年提出了类型论,它是关于集合的型的层次理论。一个集合 x 能够是一个集合 y 的元素当且仅当 y 的层次恰比 x 的层次多 1,因此再不能讲“所有集合的集合……”。类型论包括简单类型论和分支类型论两部分。J. von Neumann 于 1925 年给出了把“类”作为合法数学对象的类公理。经过 P. Bernays 和 K. Gödel 的改进与完善,又形成了另一个著名的

集合论公理系统,即 GB 公理系统。这些公理系统都能够描述朴素集合论的丰富内容,建立起朴素集合论的已有定理,排除朴素集合论中出现的悖论,并为解决集合论未解决的问题,特别是连续系统假设问题提供方便。

现在,公理集合论多采用形式化方法,利用数理逻辑的一阶谓词演算系统,建立起一阶集合论形式语言,把公理集合论的公理、定理和命题表示为该形式语言的合式公式(简称公式),从而便于用数学方法进行严格的推演与论证。因此,公理集合论不外乎是朴素集合论的严格处理。下面介绍最引人注目的 ZF 公理系统,有时为了强调选择公理,又把它称为 ZFC 公理系统。这个公理系统包括以下几条公理:

(1) 外延公理

$$\forall A \forall B (\forall x (x \in A \leftrightarrow x \in B) \rightarrow A = B)$$

两集合 A 与 B 相等是指它们的元素完全相同,并记为 $A = B$ 。把 $\forall x (x \in A \rightarrow x \in B)$ 简写为 $A \subseteq B$,并称 A 为 B 的子集。

(2) 空集公理

$$\exists x \forall y (y \notin x)$$

存在一个不含任何元素的集合,称为空集。根据外延公理,空集是惟一的,并用 \emptyset 表示。

(3) 无序对公理

$$\forall A \forall B \exists C \forall x (x \in C \leftrightarrow x = A \vee x = B)$$

对任二集合 A, B ,都有一个恰以 A, B 为元素的集合,称为 A 与 B 的无序对,并记为 $\{A, B\}$ 。记号 $\{A\}$ 表示 $\{A, A\}$,并把 A, B 的有序对 $\langle A, B \rangle$ 定义为集合 $\{\{A\}, \{A, B\}\}$ 。

(4) 并集公理

$$\forall A \exists B \forall x (x \in B \leftrightarrow \exists X (x \in X \wedge X \in A))$$

对任意集合 A ,都有一个恰以 A 的元素为元素的集,称为 A 的并集,并记为 $\cup A$ 。当 $A = \{X_1, \dots, X_n\}$ 时,常把 $\cup A$ 表示为 $X_1 \cup \dots \cup X_n$ 或 $\bigcup_{i=1}^n X_i$ 。

(5) 幂集公理

$$\forall A \exists X \forall B (B \in X \leftrightarrow B \subseteq A)$$

对任意集合 A ,都有一个恰以 A 的子集为元素之集合,称为 A 的幂集,并记为 $\mathcal{P}(A)$ 或 2^A 。

(6) 无穷公理

$$\exists x (\emptyset \in x \wedge \forall y (y \in x \rightarrow y \cup \{y\} \in x))$$

存在含有无穷多个元素的集合。

(7) 分离公理(又称子集公理)

$$\forall x \exists y \forall z (z \in y \leftrightarrow z \in x \wedge A(z))$$

其中 $A(z)$ 为 ZF 公式。

这不是一条公理,而是一个公理模式。当所涉及的对象都是某个大集合的元素时,就可以使用概括原理而获得该大集合一个子集。若 A 为任意非空集合,则由分离公理知道, $\forall A \exists B \forall x (x \in B \leftrightarrow \forall y (y \in A \rightarrow x \in y))$ 必定义一个集合,称为 A 的交集,并记为 $\cap A$ 。当 $A = \{X_1, \dots, X_n\}$ 且 $n \geq 1$ 时,常把 $\cap A$ 表示为 $X_1 \cap \dots \cap X_n$ 或 $\bigcap_{i=1}^n X_i$ 。

(8) 替换公理

$$\forall x \exists ! y A(x, y) \rightarrow \forall S_1 \exists S_2 \forall u (u \in S_2 \leftrightarrow \exists z (z \in S_1 \wedge A(z, u)))$$

其中 $\exists ! y A(x, y)$ 为 $\exists y A(x, y) \wedge \forall z (A(x, z) \rightarrow z = y)$ 的缩写。

这也不是一条公理,而是一条公理模式。设 $A(x, y)$ 为任意公式,若对任意集合 x 都有惟一的集合 y 使 $A(x, y)$ 成立,则对任意集合 S_1 都有集合 S_2 使 $S_2 = \{u \mid \text{有 } z \in S_1 \text{ 使 } A(z, u) \text{ 真}\}$ 。即 S_2 为恰由 S_1 中各元素经 $A(x, y)$ 对应的值组成的集合。

(9) 正则公理

$$\forall x (x \neq \emptyset \rightarrow \exists y (y \in x \wedge x \cap y = \emptyset))$$

对任意非空集合 x , 皆有集合 y 使 $y \in x$ 且 $x \cap y = \emptyset$ 。

正则公理说明集合与其元素之间具有某种层次关系。

(10) 选择公理

$$\forall x (x \neq \emptyset \rightarrow \exists f \forall y (y \in x \wedge y \neq \emptyset \rightarrow f(y) \in y))$$

对任意非空集合 x , 都有一个函数(称为选择函数) f , 使得当 $y \in x$ 且 $y \neq \emptyset$ 时, 有 $f(y) \in y$ 。

这个公理系统并不是独立的,如分离公理就可由其他公理推出。但由于历史原因及其使用的方便

性,还是把它列入了。

ZF 公理集合论不是完备的,因为 G. Peano 算术公理都是 ZF 公理集合论的定理,故由 G. Gödel 第二不完备性定理知道,它一定是不完备的。

参考文献

1. Takeuti G, Zaring W M. Introduction to Axiomatic Set Theory. 1982
2. Enderton H B. Elements of Set Theory. New York: Academic Press, 1977

(王兵山 张景文 吕义忠)

jijhe yunsuan

集合运算 (operations of sets) 从已知集合获得新集合的常用方法。

在讨论某类问题时,通常有一个含有所涉及的全部元素之固定集合。称为全集或空间,常用 U 表示,其他集合全是 U 的子集。假定 A 与 B 为集合。

并 A 与 B 的并集为集合 $\{x \mid x \in A \text{ 或 } x \in B\}$, 记为 $A \cup B$ 。

交 A 与 B 的交集为集合 $\{x \mid x \in A \text{ 且 } x \in B\}$, 记为 $A \cap B$ 。

差 A 与 B 的差集为集合 $\{x \mid x \in A \text{ 且 } x \notin B\}$, 记为 $A - B$ 或 $A \setminus B$ 。

补 A 的补集为集合 $U - A$, 记为 $\sim A$ 。

对称差 A 与 B 的对称差集为集合 $(A \cup B) - (A \cap B)$, 记为 $A \oplus B$ 。

如果 $A \cap B = \emptyset$, 则称 A 与 B 不相交。

上述 5 种集合运算,可用图 1 所示的文氏图直观地表示,图中阴影部分为运算结果。

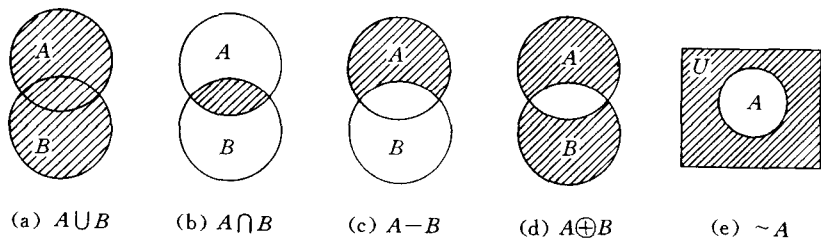


图 1 集合运算的文氏图

例 1 设 $U = \{2, 3, 5, 7, 11, 13\}$, $A = \{2, 5, 7\}$, $B = \{2, 3, 7, 11\}$, 则

$$A \cup B = \{2, 3, 5, 7, 11\}.$$

$$A \cap B = \{2, 7\}$$

$$A - B = \{5\}$$

$$A \oplus B = \{3, 5, 11\}$$

$$\sim A = \{3, 11, 13\}$$

关于集合运算 \cup , \cap 和 $-$, 有以下基本定律:

幂等律

$$A \cup A = A$$

$$A \cap A = A$$

交换律

$$A \cup B = B \cup A$$

$$A \cap B = B \cap A$$

结合律

$$(A \cup B) \cup C = A \cup (B \cup C)$$

$$(A \cap B) \cap C = A \cap (B \cap C)$$

分配律

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

同一律

$$A \cup \emptyset = A$$

$$A \cap U = A$$

零律

$$A \cup U = U$$

$$A \cap \emptyset = \emptyset$$

互补律

$$A \cup \sim A = U$$

$$A \cap \sim A = \emptyset$$

吸收律

$$A \cup (A \cap B) = A$$

$$A \cap (A \cup B) = A$$

对偶律

$$\sim(A \cup B) = \sim A \cap \sim B$$

$$\sim(A \cap B) = \sim A \cup \sim B$$

对合律

$$\sim(\sim A) = A$$

关于集合的并和交两种运算,还可做如下推广。

广义并 若 A 的广义并为集合 $\{x\}$ 有集合 $S \in A$ 使 $x \in S$, 记为 $\cup A$ 。

当 $A = \{A_1, \dots, A_n\}$ 且 A_1, \dots, A_n 均为集合时, 则把 $\cup A$ 记为 $A_1 \cup A_2 \cup \dots \cup A_n$ 或 $\bigcup_{i=1}^n A_i$ 。

广义交 若 $A \neq \emptyset$, 则 A 的广义交为集合 $\{x\}$ 若 $S \in A$, 则 $x \in S$, 记为 $\cap A$ 。

当 $A = \{A_1, \dots, A_n\}$ 且 A_1, \dots, A_n 均为集合时, 则把 $\cap A$ 记为 $A_1 \cap A_2 \cap \dots \cap A_n$ 或 $\bigcap_{i=1}^n A_i$ 。

幂集 A 的幂集为集合 $\{S \mid S \subseteq A\}$, 记为 2^A 或 $\mathcal{P}(A)$ 。

这时显然有 $\emptyset \in 2^A$ 且 $A \in 2^A$, 而且当 $A \subseteq B$ 时还有 $2^A \subseteq 2^B$ 。如果 A 有 n 个成员, 则 2^A 恰有 2^n 个成员。若 $A = \{a, b, c\}$, 则

$$2^A = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}。$$

对任意两个元素, 可把有序偶 $\langle a, b \rangle$ 定义为集合 $\{\{a\}, \{a, b\}\}$, 由此定义可知

$$\langle a, b \rangle = \langle c, d \rangle \text{ 当且仅当 } a = c \text{ 且 } b = d。$$

笛卡儿积 A 与 B 的笛卡儿积是集合 $\{\langle a, b \rangle \mid a \in A \text{ 且 } b \in B\}$, 记为 $A \times B$ 。当 $A = B$ 时, 又把 $A \times B$ 写作 A^2 。

如果 $A = \{0, 1\}$ 且 $B = \{a, b, c\}$, 则

$$A^2 = \{\langle 0, 0 \rangle, \langle 0, 1 \rangle, \langle 1, 0 \rangle, \langle 1, 1 \rangle\}$$

$$A \times B = \{\langle 0, a \rangle, \langle 0, b \rangle, \langle 0, c \rangle,$$

$$\langle 1, a \rangle, \langle 1, b \rangle, \langle 1, c \rangle\}$$

$$B \times A = \{\langle a, 0 \rangle, \langle a, 1 \rangle, \langle b, 0 \rangle,$$

$$\langle b, 1 \rangle, \langle c, 0 \rangle, \langle c, 1 \rangle\}$$

由此可知, $A \times B \neq B \times A$, 即笛卡儿积不是可交换的。

集合 A_1, \dots, A_n 的笛卡儿积 $(\dots((A_1 \times A_2) \times A_3) \dots) \times A_n$ 简写为 $A_1 \times A_2 \times \dots \times A_n$ 或 $\prod_{i=1}^n A_i$, 其元素 $\langle \dots \langle \langle x_1, x_2 \rangle, x_3 \rangle, \dots \rangle, x_n \rangle$ 简写为 $\langle x_1, \dots, x_n \rangle$, 并称为 n 元序偶。

关于集合的笛卡儿乘积, 有以下运算定律:

$$(1) A \times B \subseteq C \times D \quad \text{当且仅当 } A \subseteq C \text{ 且 } B \subseteq D$$

$$(2) A \times B = C \times D \quad \text{当且仅当 } A = C \text{ 且 } B = D$$

$$(3) A \times (B \cup C) = (A \times B) \cup (A \times C)$$

$$(A \cup B) \times C = (A \times C) \cup (B \times C)$$

$$(4) A \times (B \cap C) = (A \times B) \cap (A \times C)$$

$$(A \cap B) \times C = (A \times C) \cap (B \times C)$$

$$(5) A \times (B - C) = (A \times B) - (A \times C)$$

$$(A - B) \times C = (A \times C) - (B \times C)$$

(王兵山)

jiquanshi jisuan

集群式计算 (cluster computing) 将高档微机或者工作站, 通过系统级网络或者局域网连接起来, 在通用产品或者免费的操作系统与工具系统的支持下实现的高性能计算。集群式计算也可用于高性能服务器, 它是一种高性能、低成本的计算形式。集群式计算系统又称为机群计算系统。

集群式计算起源于美国加州大学 Berkeley 分校的 NOW, 当时称为计算机簇 (参见计算机簇)。NOW 的思想是把校园中用网络连接起来的工作站的空余计算能力集中利用起来, 而集群式计算技术更着重于高性能计算。

高性能 CPU 和高速互联网的普及是集群式

计算出现的硬件基础。根据摩尔定理, CPU 的集成度大约每 18 个月翻一番, 其计算性能也成倍地提高, 而达到相同性能所需要的成本大幅度降低。这样, 采用常见的通用 CPU 就能提供很高的计算能力。在通信技术方面, 通信延迟迅速缩短, 通信带宽成倍提高, 在高效通信的基础上, 就为多台计算机之间的并行计算提供了可能。在软件方面, 大量的功能越来越强, 性能越来越高的免费操作系统与通信软件的出现, 使得多机并行变得很容易实现, 不必从头开发相应的并行软件与系统。因此, 集群式计算这种以相对独立的计算结点(独立的 CPU, 独立的内存, 独立的操作系统)为单位的并行计算方式便得以成长并迅速普及开来。

规模较大的集群式系统采用的操作系统主要是 Linux, 而并行程序设计环境主要是 MPI, 互联网络主要是 Myrinet, Infiniband 和 Fast Ethernet。集群式计算的关键技术是减少消息传递的通信开销、建立并行程序设计环境以及开发集群管理系统与相关工具。

集群式计算除了高性能、低成本的优势之外, 还有研制周期短的优点, 集群式计算系统的建造主要是采用现成的与通用的或已商品化的软件和硬件, 即所谓的 COTS 方式。这样在大大降低研制成本的同时, 也大幅度缩短了研制的周期。而且可以采用最新的软硬件技术以提高集群式计算系统的性能。

集群式计算系统还有一个优点就是容易实现硬件系统的扩展, 系统可大可小。其缺点是不能保证系统的实际性能与系统的规模呈同步线性增长。

由于集群式计算系统的性能价格比高, 且容易实现, 它将是高性能计算的一种常见的形式。

参考文献

1. Anderson T E, Culler D E, Patterson D. A Case for NOW (Networks of Workstations). IEEE Micro, 1995, 15 (Issue) 1: 54 ~ 64
2. Gregory F Pfister. In Search of Clusters (Second Edition). Prentice Hall, 1998 (都志辉)

jisan kongzhi xitong

集散控制系统 (distributed control system, DCS) 利用计算机技术、信号处理技术、控制技术、网络通信技术和人机接口技术等对生产过程进行分散控制, 集中监视、操作和信息管理的分布式计算机控制系统。它又称分散控制系统。

集散控制系统在体系上是一种分层分布的多处

理机结构。它实现地理上和功能上的分散控制, 同时通过高速通信网络把各个分散点的有用信息集中起来, 进行集中的监视、操作和信息管理, 既克服了集中控制带来的危险性高度集中的问题, 又避免了一般控制装置分散在各设备现场操作困难且对信息无法统一管理的弱点。它不仅提高了系统的可靠性, 同时使系统扩展灵活, 减轻了操作人员的劳动强度, 减少了运行维护量, 可实现连续控制、顺序控制、批量控制等不同的控制功能, 甚至可具备实现预测控制、最优控制等先进控制的编程能力, 因此适合现代工业大生产的要求。它能完全替代单元式常规控制仪表, 在工业控制领域获得极其广泛的应用, 是当前炼油、化工、冶金、制药、电力等流程化工业进行计算机实时控制和集中管理的主流设备。

集散控制系统的出现是工业控制历史上的一个里程碑。自 1975 年美国 Honeywell 公司成功地推出世界上第一套集散控制系统 TDC-2000 以来, 它已经经历了二十多年的发展, 产品已走向成熟, 已经从初级的局部网络系统扩展成大型的分层网络系统。它的控制运算功能、应用规模、系统可靠性、信息处理能力等已有很大的提高, 产品更加标准化、综合化、开放化和现场级的智能化, 基于网络的控制系统规模更加灵活可变。在保证实时性的前提下, 系统操作站和控制站的数量可以从几个到几十个, 可满足从几个回路、几十个输入输出(I/O)信息量到上千个控制回路、上万个 I/O 信息量的用户应用要求。全世界已有数十家公司生产的数万套集散控制系统(DCS)在各种各样的工业企业运行, 取得了巨大的经济效益。

集散控制系统的组成 集散控制系统(DCS)的基本组成包括现场控制站、过程控制网络、操作员站、工程师站以及过程数据服务器等, 如图 1 所示。

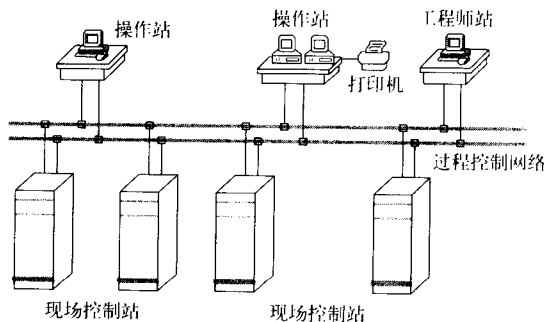


图 1 集散控制系统基本组成

(1) 过程控制网络 一类适合于工业现场环境使用的高速通信网络,它一般采用开放性和实时性较好的通信协议(如符合 IEEE 802.3 的工业以太网协议)。为了保证控制系统的高可靠性、安全性,适应现场分散安装的特点,一般采用有一定安装间距、不同敷设方式的双冗余网络。

(2) 工程师站 是配有组态工具软件和系统管理软件的微机。它为专业工程技术人员而设计,负责 DCS 的系统生成、参数设定、软件、硬件的功能“组态”,控制算法编程,系统自身的维护和管理等。在许多系统中,工程师站还能提供用户进一步开发的高级语言软件,可完成有关工艺参数的复杂运算和数据分析工作,进行用组态软件无法实现的高级控制算法编程。

(3) 操作员站 是操作工人完成过程监视和操作的人机接口设备,安装有专用的人机接口应用软件(HMI),主要用于以分级显示的画面(如总貌、分组、回路、动态工艺流程图等)监视生产过程运行的参数、状态和趋向,操纵生产过程,并可进行工艺数据的转储和打印等。

(4) 现场控制站 实现信号的采集、调理、运算及控制等功能。它常有数十种控制算法,可通过工程师站软件组态的方式,灵活地满足不同的复杂控制要求。

集散控制系统的软件 DCS 较有特色的功能软件包括:

(1) 组态工具软件 包括系统组态、过程控制组态、画面组态、报表组态等。所谓“组态”,就是确定一些软件模块的连接关系,以生成相应的各种应用软件,使用户易于修改或制定新的控制方案。

(2) 实时数据库 是应用于过程控制的实时信息交互的数据库。它必须满足实时控制要求的快速定期存储、实时处理、快速检索以及能长时间连续工作和信息安全的要求。处理的数据包括用户应用系统的组态信息、实时过程数据、过程数据报表、报警信息、历史数据等。组态生成的控制软件和人机接口软件都基于实时数据库运行。

(3) 实时控制软件 它是根据用户在工程师站的软件组态,经编译下载到现场控制站的主控制卡上的嵌入式软件,在系统内具有最高的实时性要求,运行周期一般为 0.01 ~ 1s。

集散控制系统的设计 为适应恶劣的工业现场环境(温度、湿度、电磁干扰、灰尘、腐蚀性气体、雷电、电网电压波动等)和控制要求,集散控制系统常

采取以下措施:

(1) 高可靠性设计 DCS 的现场控制站和过程控制网络设备作为工业现场的电子设备,在电磁兼容性、机械结构、温、湿度等环境指标上都应满足工业现场的要求。如采用所谓“三重隔离”(即通信接口隔离、电源隔离、I/O 通道隔离),更小的接地电阻,更高的隔离电压指标,固件密封等。此外,常要求有自诊断、自恢复或容错能力,并能容许对故障模板带电插拔,结合冗余设计,显著减少维护时间。

(2) “例外报告”的通信方法 即现场控制站仅仅将信号变化量超过一定范围的数据才加以发送,用以减少网络负荷。

(3) 高精度的信号快速采样 为了精确地反映并控制现场生产过程的工艺状态和产品质量,系统 I/O 采集卡件和输出卡件一般要具备 0.1% ~ 0.5% 的精度,以及 100 ms 到 1s 之间的采集速度。

(4) 输入输出(I/O)通道的本质安全性能 对于石油、化工等场合的应用,往往有易燃、易爆的气体或粉尘,因此对现场电气信号能量的泄放有严格的本质安全要求,输入输出通道应按本质安全要求进行设计。

集散控制系统的发展方向 DCS 的发展方向主要为

(1) DCS 将向现场总线控制系统(FCS)过渡 现场总线是一种双向、串行、多节点的全数字通信系统。现场总线技术的应用可以增加传输信息的种类,提高传输速度和可靠性,减少布线费用。现场控制单元具有更好的稳定性和精度,能够实现更复杂的算法,更加分散化,实现更彻底的开放化。

(2) 人机接口(HMI)软件的通用化和接口的统一 制定统一的软件层面的数据接口,可以提高硬件和软件的兼容能力,为客户提供更好的系统集成能力。OPC 软件协议的广泛使用使 HMI 软件的通用性大为提高,增加了工厂的综合自动化水平。

(3) 以太网向现场层延伸 工业以太网摆脱了主从式或令牌总线式的传统工业网络通信模式,采用端到端的高速数据交换,统一解决了工业网络的纵向分层和横向远程的系统问题,使信息可以从设备传感器到管理层办公桌面完全集成。以太网优越的性能、低廉的成本和高度的开放性已受到广大工业用户的关注和欢迎。

参考文献

1. 王锦标,方崇智. 过程计算机控制系统. 北京:清华大学出版社,1992

2. 王慧主编. 计算机控制系统. 北京: 化学工业出版社, 2000
(黄文君 王为民)

jixianqi

集线器 (hub) 在局域网中, 作为中枢以连接各类结点形成星状结构的网络设备。图 1 示出在以太网 10 BASE T 媒体规范中, 集线器连接个人计算机 (PC) 和服务器的情况。从图中可见, 集线器之间能互相连接以扩展结点数和延伸距离。

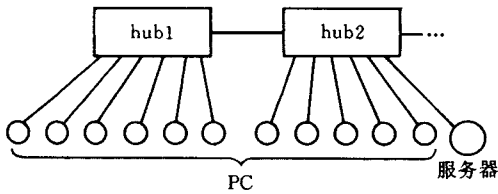


图 1 10 BASE T 的连接

20 世纪 90 年代以来, 集线器技术及其产品发展迅速, 在最基本的连线集线器基础上发展成带有中央处理器的**智能集线器**。智能集线器除了具有连线集线器的全部功能外, 依靠网络管理软件可对整个网络进行管理。以智能集线器为核心可发展成**叠堆式集线器**。图 2 中示出由 6 个集线器堆积而成的叠堆式集线器, 以 1 个智能集线器 (hub 1) 连接 5 个与之配套的非智能集线器 (hub 2 ~ hub 6)。它们共

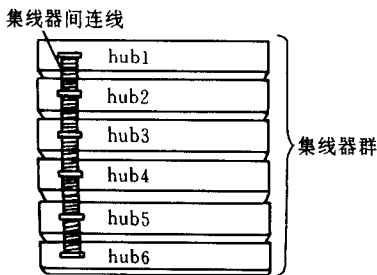


图 2 叠堆式集线器

用一个网络管理软件, 被管理的结点数可扩展 5 倍。

不仅以太网, 其他局域网, 如令牌环网、光纤分布式数据接口, 也可使用集线器来组成星状结构的网络。

箱体式集线器是一种功能更强、结构更复杂的集线器。在一个机箱中, 可安装多个模块 (如图 3 所示), 其中一些模块可以是以太网集线器, 也可以是令牌环 (token ring) 或光纤分布式数据接口 (FDDI) 集

线器。在集线器模块之间可安装网间互连设备 (例如网桥或路由器) 模块以实现网络间的互连。在箱体中也可安装**广域网** (例如 X.25 公用数据网) 接口模块以实现整个系统的广域连接。

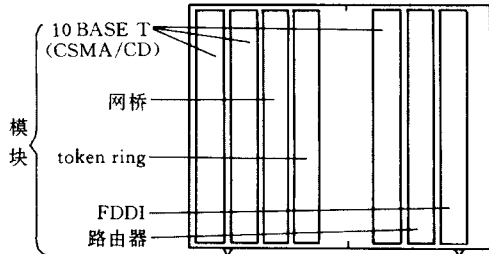


图 3 箱体式集线器

集线器与结点之间的连线常选用双扭线, 也可以选用光缆。

集线器技术及其产品的发展, 使得网络系统具有布线灵活方便, 可靠性高, 易管理维护, 易扩展等特点。

近几年来, 各种集线器设备大量涌现, 如收发器集线器、路由器集线器及**交换式集线器**等。以太网交换式集线器所组成的网络系统不仅具有上述全部特点, 而且使得整个网络系统的频宽大大增加, 还能实现系统内网络的分段和隔离。 (张公忠)

jizhongqi

集中器 (concentrator) 在数据传输中, 一种能在若干条低速线路 (通常是异步的) 与一条或多条高速线路 (通常是同步的) 之间提供数据通信的功能装置。

集中器一般具有数据缓冲能力, 用于吸收高峰负载时的数据, 在持续高峰负载情况下也可用暂时中止方式对某些设备的传输进行调节。

集中器可以是面向字符的, 也可以是面向报文的。它能支持多种不同的设备。在**计算机网络**中, 集中器常处在远离计算机且低速终端比较集中的地方, 以便把大量的低速数据汇集起来送到计算机中去, 其目的是提高线路的使用效率, 降低传输成本。 (过介望)

jizhongshi wangluo guanli

集中式网络管理 (centralized network management) 把网络管理的各个要素集中在网内一

个系统上的一种计算机网络管理。网络的公共管理信息要素主要指网络管理功能和网络管理信息等(参见 OSI 管理体系结构)。

集中式网络管理系统中有一个中心管理系统,所有的管理功能和管理信息都集中在这里。通过这个中心管理系统可向网络管理操作员提供管理功能,以便对网络中有关资源进行监视和控制。

集中式网络管理比较适合于较小的计算机网络的管理。它的优点是管理方便,这是由于全部网络管理功能和管理信息都集中在一个系统上的缘故。但是,如果某个网络资源和中心管理系统之间的连接发生故障,那么,对这个资源的管理就不能进行。并且,如果中心管理系统发生故障,就会造成整个网络管理功能的瘫痪。

参考文献

Subramanian M. Network Management: Principles and Practice. Addison - Wesley, 1999 (钱松荣)

jihe zaoxing

几何造型(geometric modeling) 采用计算机进行几何形体建模的技术。几何造型技术是计算机辅助设计、分析、制造以及计算机图形学的基础。

几何造型技术自 20 世纪 60 年代出现以来,经过几十年的发展、进化,已日臻成熟,内容十分丰富。几何造型从总体上可以分为线框造型、曲面造型和实体造型 3 种。

线框造型最简单,采用顶点和棱边表示几何形体(称为线框模型),通过交互输入几何形体的顶点和棱边进行几何造型。线框模型的特点是表示简单,算法高效。然而,由于采用线框模型表示三维物体时几何信息不完备,无法支持对三维物体进行剖面图生成、隐藏线消除、面面求交等操作,因此,线框模型只能用作三维形体的辅助表示。

曲面造型是应精确描述飞机、船舶、汽车的流线外形的需要而发展起来的,其核心内容是复杂曲面的数字化定义和表示方法。为曲面造型奠定理论基础的杰出先驱是法国的 Pierre Bézier 和美国的 Steven Coons。Steven Coons 提出了将分段曲线和分片曲面拼合成任意复杂曲线、曲面的一般处理方法。Bézier 则构想了另一种更加直观的曲线、曲面设计方法,即用折线来勾画曲线、曲面形状,生成 Bézier 曲线、曲面的方法。1973 年 R. F. Riesenfeld 将 Bézier 曲面中的 Bernstein 基函数替换成 B 样条基函数,形成了 B 样条曲面。以后又进一步发展为非均

匀有理 B 样条(NURBS)曲面。由于 NURBS 曲面能够统一表示平面、二次曲面、Bézier 曲面和 B 样条曲面,它已成为 CAD 系统中国际公认的产品外形定义方法。上述曲面造型方法所生成的曲面均为参数曲面。这些方法统称为自由曲面造型方法,其突出的优点是曲面构造直观,修改方便,能够构造任意复杂曲面等。隐式曲面造型是另一种曲面造型方法。由于代数隐式曲面在表示某些种类的几何形体时更加方便、准确,如对计算机动画中的人体和动物的造型,因此隐式曲面造型近年来发展较快,其中具有代表性的是元球方法。曲面造型从造型效果上又可分为插值、拟合和逼近 3 种。

实体造型旨在建立三维物体的完备几何模型,即实体模型,以支持 CAD/CAE/CAM 的一切方面。实体造型技术产生于 20 世纪 60 年代末至 70 年代初,英国的 Ian Braid 和美国的 A. A. G. Requicha 是该技术的主要奠基人。实体模型主要包括边界表示法(B-rep)、体素构造表示法和空间枚举表示法。实体造型操作主要有拼合操作、扫成操作、欧拉操作、局部操作等。传统的实体造型经过 10 多年的演变进一步发展为基于参数化和特征的实体造型。所谓“基于特征的实体造型”是指用带有特定工程语义的形状单元、形状特征取代传统的基本体素进行实体造型的方法。由于形状特征为本行业的专业人员所熟悉,并带有特定的设计知识和加工知识,因此基于特征的实体造型更符合设计人员的设计习惯,而且能够生成包括实体模型和特征模型在内的信息更加充分的产品模型。参数化实体造型方法的特点在于将几何约束引入实体模型中,形成以尺寸为参数的参数化模型,以有效地支持变动设计。采用参数化实体造型,在构造出几何形体的参数化模型后,可以通过修改当前几何形体的尺寸参数值自动生成与新的尺寸相匹配的几何形体,从而显著提高变动设计的效率。

几何造型技术的应用范围很广,包括 CAD、CAE/CAM、计算机图形学、虚拟现实、可视化、计算机动画、地理信息系统、计算机视觉、计算机仿真等学科领域。

参考文献

1. M E Mortenson. Geometric Modeling. New York: John Wiley & Sons, 1985

2. 唐荣锡,汪嘉业,彭群生,汪国昭. 计算机图形学教程(修订版). 北京:科学出版社,2002

(高曙明 刘玉生)

jiliang yuyanxue

计量语言学 (quantitative linguistics) 用数学方法(概率论、数理统计、随机过程、模糊逻辑、集合论、信息论、函数论等)研究各级语言单位的出现频率和分布规律,以便建立基于统计学特性的语言模型的学科。

当人们对语言和文本进行观察时,记下不同层次上各类语言单位(如音素、音位、字及其笔画、部件、结构方式、词、词类、词义、短语结构、句型等)的出现次数,并在所得的频率数据之间寻找相互关系,就可能发现某种支配这些单位在使用时的统计规律。语音、文字、词汇和语法的许多方面都曾经运用这种方式进行观察和研究,并获得了有实用价值的统计法则。这说明人们的语言行为严格地遵循着某些统计期望。调查表明某些统计规律甚至普遍适用于所有语言。美国哲学家 G. Zipf 提出的 Zipf 定律就是一条与语种无关的普遍法则。该定律认为在一种语言中每个词型在按降序排列的词频表中的秩(又称词级) r 和它的出现频率 f 之间存在简单的反比关系。这一关系与文本的作者、主题或其他语言因素无关。我国学者在进行现代汉语词频统计时论述了这条定律同样适用于汉语。只是在频率最高和最低的那些词型上略有差异。Zipf 通过观察还发现,一个词型的长度与其出现频率成反比关系。换句话说讲,越是频繁使用的词,其语音越简单且词长越短。现代汉语的词频统计同样验证了这条统计规律:在全部词型中,汉语双音节词占 73.6%,单音节词只占 12%;但就词次而言,单音节词占 64%,双音节词占 34%,其余三音节以上词的词次总和不到 2%。G. Zipf 把这条统计规律解释为语言使用中的最省力原理。字频和词频统计在语文教学、字典和词典编纂以及作家的文章风格研究中所起的作用已经为人们所熟知;它们对设计汉字编码(键盘)输入方案(参见**汉字编码(键盘)输入方法**)、信息交换用编码字符集、**汉字识别**、语音识别、自动分词和文本校对等领域的贡献也是中文信息处理界有目共睹的事实。随着**自然语言处理**、**机器翻译**、全文检索、信息摘录、自动文摘和文献分类等研究的深入,为了解决语言各层次上出现的大量歧义现象,近年来研究人员更加重视从大规模真实文本语料库中用统计方法直接获取各类语言知识和相应的排歧规则,形成了**语料库语言学**这一新学科。其中著名的研究成果有词性自动标注、词义排歧、英语中介词短语的语义指向、概率型上下文无关语法(PCFG)、语音识别中

的隐马尔可夫模型(HMM)、自动(机械)文摘、文献分类等等。

这个学科的主要研究内容包括:①运用精密的数学方法研究语言和文本的计量特性;②把计量语言学的方法、模型和研究成果应用于自然语言处理、机器翻译、信息检索、语言教学和文档生成等实用性课题;③把自然科学、经济学和心理学等学科中的科学方法和模型引入语言文字学。

国际学术界成立了国际计量语言学学会(IQLA),定期举行国际学术会议,并出版学报 *Journal of Quantitative Linguistics*。

参考文献

1. 陈原. 现代汉语定量分析. 上海:上海教育出版社,1989
2. E Charniak. *Statistical Language Learning*. Cambridge: MIT Press, 1993
3. C Butler. *Statistics in Linguistics*. UK: Basil Blackwell, 1985

(黄昌宁)

jishiqi

计时器 (timer) 按一定的时间间隔来改变其内容用以度量时间的一种寄存器。又称**定时器**。

在计算机中,计时器的动作一般是在程序控制的基础上进行的,它在经过预定的时间间隔后所发出的信号常用以产生程序中断。计时器时间间隔的大小也可由程序来设置。计时器主要用于报告作业各部分的执行时间和日期,检查例行程序各段的定时及启动某些事件等。尤其是在多道程序系统中,为了保持每个作业的间隔定时,必须使用计时器来控制中央处理器指派给某个特定作业所经过的时间。

计时器可以是计算机控制器中的一个特殊的寄存器,也可以利用内存存储器中某个字来实现,通常该字处于内存存储器的低端,只供计时,不作它用。

(过介望)

jisuan fuzaxing lilun

计算复杂性理论 (computational complexity theory) 用数学方法研究各类问题的计算复杂性的学科。它研究各种可计算问题在计算过程中资源(如时间、空间等)的耗费情况,以及在不同计算模型下,使用不同类型资源和不同数量的资源时,各类问题复杂性的本质特性和相互关系。具体说,第

一个层次是具体复杂性,它研究各种问题的时间和空间复杂度。这一类研究通常包括在算法设计与分析学科分支中。第二个层次是问题复杂度的上界和下界研究。虽然这种研究仍是针对具体问题的,但它包含了问题复杂性的一些本质特性,也采用了一些更具广泛意义的技术与方法。再上一层是研究语言复杂性类,研究它们的时间、空间以及时空折合等,研究确定性与非确定性对复杂度的影响。最后,研究完全性、相对性和相似性。

资源与复杂度函数 最重要的资源是计算过程所占的时间和所用的空间。资源的精确定义与所采用的计算模型有关。其他资源尚有图灵机的巡回,布尔线路的大小、深度和宽度,向量机器模型的并行时间等。计算过程所耗费的资源与问题的规模有关。例如, n 个数的排序, n 越大,所需的时间自然越多。比较好的排序算法的时间是 $cn \log n$ (c 是一常数)。在计算复杂性理论中,算法所需的时间和空间都是问题规模 n 的函数 $T(n)$ 和 $S(n)$,它们分别称为时间复杂度函数和空间复杂度函数。当 n 确定以后,可能还有多种不同的实例,不同实例的计算时间仍可能不同。如果选用所有规模为 n 的实例所用的最长时间作为 $T(n)$,则称之为最坏情况时间复杂度。如果选用各实例所用时间的平均值作为 $T(n)$,则称之为平均时间复杂度。大多数情况下,采用最坏情况复杂度。类似地可定义空间复杂度。

计算模型 最基本的模型是图灵机。作为时间复杂性分析的模型是多带图灵机。它有 k 条双向无限带,其基本动作是:根据有限控制器的状态和 k 条工作带带头读到的 k 个字符,执行三个操作,即 k 个带头各自写下一个字符,改变机器状态, k 个带头独立地左移或右移一格。对于每个长度为 n 的输入,图灵机 M 在停机前最多做 $T(n)$ 个动作,就称 M (及其对应的算法)的时间复杂度为 $T(n)$ 。作为空间复杂性分析的模型是离线图灵机。它有一条只读输入带和 k 条半无穷存储带。其优点是可以进一步考虑比线性空间更小的空间耗费,例如对数空间。对于每个长度为 n 的输入,图灵机 M 在任一条存储带上至多扫视 $S(n)$ 个方格,就称 M (及其对应的算法)的空间复杂度为 $S(n)$ 。

其他计算模型还有随机存取机(RAM),非确定图灵机、Oracle 机器、布尔线路以及并行计算模型P-RAM和向量机器模型等。

上界与下界 研究计算复杂性的一个重要目的是要为各种问题寻求最优算法。这就要求知道问题

的固有复杂度。目前对大多数问题而言,还只知道它最多需要多少、至少需要多少某种资源。以时间为例,前者称为该问题时间复杂度的上界,后者称为下界。上界越小越好,下界越大越好。一旦对某问题,其上、下界相等了,就知道了该问题的固有复杂度,具有这种复杂度的算法自然是最优算法。上下界研究是针对单个具体问题的。

寻找问题的上界,没有固定方法。找到问题的一个算法,若其时间为 $T(n)$,则 $T(n)$ 即为该问题时间复杂度的一个上界,因为这表明,解决问题用 $T(n)$ 时间就足够了。更小的上界,取决于设计更好的算法。 n 点离散傅里叶变换的时间上界为 $O(n \log n)$ 。两个 n 维矩阵的乘法,通常需要 $O(n^3)$ 时间。用分治法设计的一个算法,使时间上界降到 $O(n^{2.81})$ 。经过一系列工作,时间上界已降到 $O(n^{2.376})$ 。

寻找问题的下界,则困难得多。因为此时必须证明,该问题的一切算法(包括已知的和未知的),都至少需要这么多资源。证明下界的常用方法有计数论证法和归约方法。目前关于下界的结果,多集中在语言理论和逻辑理论中。下面是两个通常问题的下界结果: n 个数的排序,其基于数的比较的时间复杂度至少是 $n \log n$;为了在图灵机上判断一个长度为 n 的数是否是某数的完全平方,其时间和空间的乘积至少正比于 n^2 。

语言复杂性类 语言(参见形式语言理论)是字符串的集合。如果对于一个输入字符串,图灵机操作到最后,进入某个接受状态并停机,就称这个图灵机接受该字符串。如果一个图灵机接受且只接受某个语言中的字符串,就称这个图灵机识别该语言。语言可以按识别它们时所耗费的时间,分成不同的语言复杂性类,例如, $DTIME(n)$, $DTIME(n^2)$, $DTIME(n^3)$, ..., $DTIME(2^n)$, ...。这里 $DTIME(f(n))$ 表示确定型图灵机在 $f(n)$ 时间内所能识别的所有语言组成的类。类似地,语言也可按所耗费的空分分成各种语言复杂性类,例如 $DSPACE(\log n)$, $DSPACE(n)$, $DSPACE(n^2)$, ..., $DSPACE(2^n)$, ...。还有两个重要的语言复杂性类P和PSPACE,它们分别是在多项式时间和多项式空间内能识别的语言组成的类。显然,P包含 $DTIME(n)$, $DTIME(n^2)$, ..., $DTIME(n^k)$, ...。P类问题(对应于P类中语言的问题,参见P类问题)都是多项式时间内可解决的问题,被公认是在计算机上现实可解的问题,解决它们的多项式时间算法被认为是现实可行的

算法。

计算复杂性理论的一些主要研究工作都集中在语言复杂性类上。这是因为各种问题的求解,各种函数的计算,都可转化成语言的识别。在计算复杂性理论中,问题由无数实例组成,实例经过编码可表成字符串,所有具有肯定回答的实例所对应的字符串就构成一个语言。因此,判定一个实例是否有肯定回答,就转化成判定一个字符串是否属于一个语言。对于函数的计算,可以在图灵机上增加一条输出带,自变量值经编码可作为输入字符串,设计图灵机时,保证当机器停机并进入接受状态时,输出带上正好写下了函数值。这样,也转化成语言识别问题。

时间谱系与空间谱系 某些基础研究工作在于建立语言复杂性类的时间和空间谱系,即要弄清楚语言复杂性类由于时间和空间等资源量的不同,是怎样一层一层扩大的。线性加速定理表明,时间和空间仅由 $f(n)$ 增加到 $kf(n)$,所识别的语言不会增加,因而语言复杂性类不会扩大。关于谱系的一些基本定理回答了时间和空间究竟要分别增长到什么程度,才会使所识别的语言有所增加。此外,还证明:任给一个资源函数 $f(n)$,总会有语言不在 $DTIME(f(n))$ (或 $DSPACE(f(n))$)中,因而时间谱系和空间谱系都是无穷的。

确定性与非确定性 一个确定型图灵机,对于其状态和所读到的字符,只有一个惟一的动作(包括改写字符、改变状态和移动带头等三个操作)。如果有多个动作可供选择,此时,由于每步都可能有几个不同的动作,这种图灵机对一个输入字符串,从开始到停机,将可能会有多种不同的动作序列,若规定只要有一个序列终止在接受状态,机器就接受这个输入字符串,则这种机器称为非确定型图灵机。已经证明,非确定型图灵机与确定型图灵机所识别的语言类是相同的。若一旦考虑有界的时间与空间,它们的识别能力需认真分析。

对于非确定型图灵机,也可按时间和空间界限,将语言分成许多语言复杂性类,形成非确定时间和空间谱系。例如, $NTIME(n)$, $NTIME(n^2)$, \dots , $NTIME(2^n)$, \dots ; $NSPACE(\log n)$, $NSPACE(n)$, \dots , $NSPACE(2^n)$, \dots 。这里也有两个重要的语言复杂性类 NP 和 NSPACE,它们分别是非确定型图灵机在多项式时间和多项式空间内识别的语言组成的类。在确定型和非确定型语言类之间,显然有下面关系: $DTIME(f(n)) \subseteq NTIME(f(n))$, $DSPACE(f(n)) \subseteq NSPACE(f(n))$ 。在时间和空间

之间有: $DTIME(f(n)) \subseteq DSPACE(f(n))$, $NTIME(f(n)) \subseteq NSPACE(f(n))$ 。有大量工作是集中在研究各种语言复杂性类之间的关系上的,研究时间与空间的关系、确定性与非确定性的关系。已经证明, $DTIME(n) \neq NTIME(n)$ 。这说明在线性时间界限内,确定型图灵机所识别的语言类要严格地小于非确定型图灵机所识别的语言类。另一方面,也已证明, $PSPACE = NSPACE$ 。这说明在多项式空间界限内,确定型图灵机和非确定型图灵机所识别的语言类又是相同的。这个结论来自一个著名的定理: $NSPACE(S(n)) \subseteq DSPACE(S^2(n))$ 。有许多问题还不明朗,例如是否有 $DTIME(n^k) \neq NTIME(n^k)$, $NTIME(n) = DSPACE(n)$ 等。在确定性和非确定性关系方面,还有一个十分著名的悬案: $NP = ? P$ (相关的还有 $NP = ? PSPACE$)。这是一个在计算机科学领域中历经 20 多年尚未解决的问题。它在理论上和实践中都有极其重要的意义,它的解决将导致一系列理论问题的解决,还将关系到计算机科学、数学、逻辑学和运筹学中一大批实际问题是否是“现实可计算的”(参见 NP 完全性理论)。

完全性 在语言复杂性类中,常常要研究类中最困难的语言(最困难的问题),完全性就是最困难性的形式表示。利用归约,可以解决两个问题的困难程度的比较问题,从而有助于完全性的定义。有各种不同形式的归约。在研究 NP 语言复杂性类的完全性时,要利用多项式时间多一归约。一个语言 L 多项式时间多一归约到语言 L_0 ,如果存在一个确定型图灵机,它在多项式时间内将一个字符串 x 转变成一个字符串 y ,并使得 x 是 L 的字符串,当且仅当 y 是 L_0 的字符串。于是¹ L 的识别问题就归结为 L_0 的识别问题,反之, L_0 体现了 L 的难度。一个语言 L 称为是 NP 完全的,如果 L 是 NP 中语言,且 NP 中任一语言都能多项式时间多一归约到 L 。类似地可以定义 P 完全性, PSPACE 完全性等。由于 NP 类中一切语言的识别都可以归结为一个 NP 完全语言的识别,因此,可以认为 NP 完全语言是 NP 类中最难于识别的语言。在诸多的完全性研究中, NP 完全性研究有特殊地位,它包括许多研究课题,是计算复杂性理论中一个重要研究领域。其研究与 $NP = ? P$ 问题有密切关系,在理论和实践两个方面,都有重要意义(参见 NP 完全性理论)。

相对性 这是从另一个角度来比较问题求解的复杂程度,它需要借助某种外部信息。研究的基本工具是 Oracle 机器。Oracle 机器包括一台图灵机,

一个 Oracle 集(直译为神灵启示集或译为外部信息源),一条特殊带和三个特殊状态 Query, Yes 和 No。每当机器进入 Query 状态时,机器就要询问 Oracle 集:特殊带上写的字符串是否是 Oracle 集中的字符串。如果是,机器进入 Yes 状态,否则进入 No 状态。然后进入不同的后续操作。整个机器动作,除上述情况外,与图灵机无异。这是一种相对化的计算,它不仅依靠机器自身的计算,还强烈地依赖于 Oracle 集合的性质。Oracle 集合可以是一个具有某种复杂度的语言,甚至也可以是一个不可计算的集合。以 A 为 Oracle 集合的确定的 Oracle 机器在多项式时间接受的语言类,记作 $P(A)$,类似地可定义 $NP(A)$ 。

利用 Oracle 机器,可以形成许多新的相对化的语言复杂性类,它们可以组成一个完整的体系——多项式谱系,这个谱系有着丰富的内容。在相对性研究中最重要一个研究结果是:存在一个集合 A ,使得 $NP(A) = P(A)$,同时,还存在另一个集合 B ,使得 $NP(B) \neq P(B)$ 。这一结果更显示了解决 $NP = P$ 问题的困难程度。

相似性 由于计算模型的不同,同一类问题所用的资源可能不尽相同。那么,这些复杂度函数之间究竟呈现什么关系呢?有人提出并行计算假设:并行机器的时间复杂度多项式相关于图灵机的空间复杂度。在全面深入研究的基础上,人们又提出相似性原理:对于同一类问题,各理想的计算模型使用本质上同样多的并行时间,本质上同样多的空间和本质上同样多的串行时间。“本质上同样多”的含义是多项式相关。对于图灵机,并行时间系指巡回。

此外,还有描述复杂性和 Kolmogorov 复杂性等研究方向。

目前,随着科学技术的发展,计算复杂性理论又进入一个新的发展阶段。日益占据重要地位的并行计算,推动了并行计算复杂性理论的发展。特别是人工智能和智能计算机的研究,由于涉及对人类智力行为的模拟,以及巨大的信息处理量,必将对计算复杂性理论提出崭新的研究课题。

参考文献

1. Hopcroft J E, Ullman J D 著. 自动机理论、语言和计算导引. 徐美瑞译. 北京: 科学出版社, 1986
2. Wagner K, Wechsung G. Computational Complexity. Dordrecht, Reidel, 1986
3. Balcazar J L, Diaz J, Gabarro J. Structural

Complexity I, II. Berlin: Springer-Verlag, 1988

4. Papadimitriou C H, Computational Complexity. Addison-Wesley Publishing Company, 1994 (徐美瑞)

jisuanji RAS jishu

计算机 RAS 技术 (computer reliability availability and serviceability) 计算机系统可靠性、可用性和可服务性 3 项技术的总称。它是研究、设计、评价计算机系统,特别是高可靠的计算机系统必不可少的重要内容。

本来广义的计算机系统可靠性就包含可用性和可维性在内。20 世纪 70 年代后期,由于可用性和可维性技术的发展,才产生了更为直观的 RAS 概念,突出了人们对可用性技术和可维性技术的关注。

计算机系统可靠性技术是与计算机系统本身的发展同步的。1951 年,第一台商品计算机 UNIVAC I 就采用了奇偶检验码,并用两个运算部件,以匹配-比较方式工作。1969 年 STAR 容错计算机研制成功,进一步促进了容错技术的发展。特别是由于大规模集成电路的发展和计算机应用的普及,人们对计算机 RAS 技术性能的要求愈来愈高。在军事、航天、金融等部门,对计算机可靠性有更苛刻的要求。

对于实用的计算机系统,由于系统规模、应用场合、环境条件的差别以及成本价格因素的影响,其 RAS 技术的投入强度和实现方法是有所不同的。

可靠性 系统在规定的工作条件下和预定的时间内持续正确运行(完成规定功能)的概率。可靠性通常用平均故障间隔时间(MTBF)来表征。

可靠性技术 即在进行系统可靠性设计中所采用的基本技术。

主要包括:

(1) 系统可靠性预计 计算机系统由各类元件组成。系统可靠性预计即根据各类元件的可靠性以及各元件之间的连接关系构成的可靠性模型作系统可靠性的预先分析计算,以预测系统是否达到可靠性指标;找出关键元件、关键路径,为改进可靠性设计提供依据。

(2) 可靠性分配 根据系统总的可靠性指标,将其分解,并对各分系统,直至器件、工艺提出相应的可靠性指标。

(3) 元件优选与筛选 在成本允许范围内,选

择高可靠元件,并进行老化筛选。

(4) 线路工程 通过对关键线路的工程试验与分析,制定工程设计规范,确定生产工艺参数,将外部干扰和内部噪声控制在容限范围内。

(5) 热设计 合理布局热源,制定冷却方案,控制元器件工作环境的温度和湿度。

(6) 容错设计 通过冗余技术消除或减轻机器校验出错或故障造成的影响。冗余技术包括硬件冗余、软件冗余和时间冗余。硬件冗余有校验纠错、双工系统、多数表决和其他复式冗余结构。软件冗余有关键数据和程序的重复存储,设置多个程序出入口等。时间冗余主要以时间为代价,通过重复执行(简称复执)消除偶发性故障的影响。复执可以在微操作、指令、程序段或整个作业等各种级别上进行。多数计算机系统往往同时采用上述多种冗余技术。

可用性 系统在规定的条件下能正常工作的概率。一般用平均利用率 A 来表示:

$$A = \frac{MTBF}{MTBF + MTTR}$$

式中 $MTTR$ 为系统的平均修复时间; $MTBF$ 为平均故障间隔时间。

如果把系统正确完成规定功能的总时间称为正常运行时间,把系统维修总时间称为故障时间,则平均利用率 A 可表示为:

$$A = \frac{\text{正常运行时间}}{\text{正常运行时间} + \text{故障时间}}$$

通常把 A 叫做系统可用率。

可用性技术 可用性技术是基于可靠性技术和可维性技术,旨在提高系统可用性所采用的技术。主要包括:

(1) 冗余结构 在系统构成上采用复式模块化冗余结构。当某一部件出现故障时,可自动或人工重构系统,由另一部件替代故障部件工作,系统降级使用,除故障部件外,其他部分仍保持正常运行。

(2) 并行维修 为故障部件建立维修环境,使维修工作与系统正常运行同时进行。

(3) 容错计算 通过多重处理或其他软硬件相结合的技术手段,消除机器故障造成的影响,使用户感觉不到机器曾经发生过故障。

可服务性 系统可服务性是系统可维护性和可修理性的总称。可维护性是指对系统运行状态进行监视、控制的支持能力以及更换易损件的方便性。

可修理性是指系统发生故障后,从查找故障原因到更换故障部件或修改故障软件使之重新投入正常运行的支持能力。一般,可服务性用平均修复时间($MTTR$)来表征。

可服务性技术 为提供方便的维修,缩短维修时间所采取的技术。主要包括:

(1) 系统控制台 集中监视机器运行状态,校错,故障报警,提供必要的软件、硬件维护以及调试支持。

(2) 自测试 在机器建立初始环境或在正常运行过程中,自动进行功能测试。

(3) 故障自动诊断 一旦机器发生故障,即自动或人工进入故障诊断,定位故障点。

(4) 连机维修 允许在正常运行或带电状态下更换故障部件。

(5) 维修培训 使维修人员熟练掌握维修技术,缩短维修时间。

综上所述,可靠性、可用性、可服务性 3 项技术各有其自身的技术含义,又相互关联。上述平均利用率的表示式集中体现它们之间量的关系。

可靠性数学模型 可靠性数学模型研究是计算机可靠性理论研究的一个重要内容。目前已有多种可靠性模型用于系统可靠性设计、分析与评估。这些模型包括:马尔可夫链、可靠性框图、故障树、乘积型排队网和串并有向无环图等。与此同时,支持多种模型的可靠性分析工具软件也日趋成熟,有代表性的工具软件包括: HARP, SURE, Heiress, SHARPE 和 Save 等。

未来发展 计算机 RAS 技术是计算机科学与技术的一个重要分支。随着超大规模集成电路技术的发展以及多机系统、大规模并行处理系统、分布式系统的发展将进一步促进 RAS 技术的研究与发展。今后,在可靠性建模,系统可靠性预计与评估,故障自动诊断,容错计算与处理,并行维修,软件可靠性与软件正确性证明等方面也将进一步得到发展与完善。功能更强的容错系统、“不停机”的计算机系统、“傻瓜”式的计算机将会出现。

参考文献

1. 猪濑·博编著. 计算机系统的高可靠性技术. 尤国峻, 肖俊远译. 北京: 国防工业出版社, 1985
2. 傅佩琛等. 计算机系统硬件软件可靠性理论及其应用. 北京: 国防工业出版社, 1990 (陆绍福)

jisuanji anquan

计算机安全 (computer security) 确保计算机系统正确、可靠、不间断运行,保护运行中的计算机系统存储、传输、处理的信息不因人为的和自然的原因遭到泄露、破坏或无法使用的机制和措施的总称。考虑到计算机安全包括系统安全运行和信息安全保护两方面的内容,计算机安全也称为**计算机信息系统安全**。鉴于当今计算机技术与网络技术相互渗透、融为一体的发展趋势,计算机安全涉及包括计算机、网络在内的计算机信息系统的安全。

发展简史

计算机安全技术与计算机技术是同步发展的技术。然而,只有在多用户计算机系统得到快速发展和普遍应用时,计算机安全才受到重视。1985年美国国防部发布了业界第一个计算机安全标准:《**可信计算机系统评估准则**》,标志着对计算机安全技术已经有了比较系统化的研究。早期的计算机安全主要考虑单机系统,且人员和环境都比较可信的情况。随着计算机网络技术的发展和应用的普及,以及计算机在信息处理方面的广泛应用,计算机、网络已成为社会生活的重要组成;网络“入侵者”的出现及其所造成的威胁极大地影响了计算机的正常使用。这些因素促使计算机安全技术快速发展。20世纪90年代中期以来,计算机安全逐步从单纯的信息保护发展为包括信息保护、运行检测、快速反应和故障恢复等内容的信息保障技术。国际标准化组织于1999年发布了最新的信息安全标准:《**信息技术 安全技术 信息技术安全性评估准则**》,从更加广泛的领域对包括计算机、网络在内的信息安全技术进行了全面描述。当今计算机安全技术的发展,已将密码技术与系统安全技术相结合,构成了完整的安全保护体系。密码系统所提供的对存储和传输信息的保密性、完整性、真实性支持,在计算机安全中起到越来越重要的作用。这些均标志着计算机安全技术的发展进入一个新的发展时期。

1994年2月18日,我国国务院发布的《**中华人民共和国计算机信息系统安全保护条例**》对计算机信息系统实行安全等级保护。1999年9月13日由国家质量技术监督局发布的《**计算机信息系统安全保护等级划分准则**》将计算机信息系统的安全划分为5个安全保护等级,从低到高依次为:用户自主保护级、系统审计保护级、安全标记保护级、结构化保护级和访问验证保护级。等级划分准则及其配套标准和相应的测评工具为我国计算机信息系统安全等

级保护制度的实施提供了强有力的支持。

计算机安全的基本内容

计算机安全的最终目标是实现对计算机系统中信息的安全保护和对提供有效信息服务的保障。计算机安全涉及十分广泛的内容,包括信息的保密性、完整性、可用性(含可控性、可操作性和不可抵赖性等)。计算机系统安全的实现,需要从保护、检测、反应、恢复等方面,通过安全功能的设置和安全保证措施的实施来达到。安全功能具体体现在实体安全、运行安全 and 信息安全等方面;安全保证则在安全管理、系统设计与实现以及安全机制自身安全等方面提供支持。

实体安全是指支持计算机信息系统运行的计算机、网络硬件设备、设施、记录媒体及其环境和辅助设施的安全。所采取的措施主要包括:计算机主机、外部设备、网络通信设备及记录媒体的安全保护,机房选址和机房内部的安全防护,供、配电系统的安全保障,防火、防水、防震及电磁防护等方面的安全措施。

运行安全是指在实体安全的基础上,计算机系统的不间断运行。所采取的措施主要包括:系统安全检测,运行安全监控,安全审计,计算机病毒防杀,备份与故障恢复以及应急计划与应急措施等。

信息安全是指对计算机系统中存储、传输和处理的信息实施有效的保护。所采取的措施主要包括:防止非法进入系统的标识和认证;确保信息交换真实性的安全鉴别;防止越权访问的标记与访问控制;确保存储数据的保密性、完整性;确保传输数据的保密性、完整性;确保处理过程中数据的完整性、一致性和对剩余信息进行保护的一系列措施。

安全管理是指对与计算机信息系统运行相关的人员、设备、设施、环境等方面的管理,以确保为实现实体安全、运行安全、信息安全所设置的安全机制和所采取的安全措施能够发挥其应有的作用。这需要通过对人员进行严格的培训,制定必要的操作规程、规章制度和有效的实施细则来保证。

计算机系统及其安全机制的设计与实现对于计算机系统的安全性能否达到确定的安全目标至关重要。因此,应从开发设计、文档设计、测试、脆弱性评定、配置管理、分发与安装、生命周期支持等方面,严格按照有关规定实施。

安全机制的自身安全对计算机系统的安全十分重要,需要从物理安全保护和运行安全保护等方面,确保安全机制不因人为的或自然的原因遭到破坏而

失效,特别对于以增强的方法实现计算机系统安全的机制,更应注重对“安全功能无效”攻击的防护。

建设安全的计算机系统的方法和步骤为:①进行风险分析,确定安全需求;②制定安全方案——确定安全策略和安全机制;③系统设计和实现,将安全策略和安全机制付诸实施;④系统测试与评估,确定实现的计算机系统是否达到所需要的安全性。

发展趋势

以计算机、网络为支撑的信息化、知识化是当今社会发展的方向。计算机的应用已经无处不在。人为的或自然的原因对计算机系统的运行和使用所造成的威胁和破坏已屡见不鲜。随着电子商务、电子政务等对更大规模、更重要的计算机应用的拓展,计算机安全将更加受到重视,计算机安全技术将得到更大的发展。以公开密钥设施(PKI)为核心的信息安全基础设施,为计算机安全技术的进一步发展提供了强有力的支持。密码技术与系统安全技术的更紧密结合,已经成为计算机安全技术的重要发展方向。

参考文献

1. 公安部计算机管理监察司. 计算机信息系统安全技术. 北京:群众出版社,1989
2. 王锡林,郭庆平,程胜利. 计算机安全. 北京:人民邮电出版社,1995
3. ISO/IEC 15408:1999 Information Technology—Security Techniques—Evaluation Criteria for IT Security (吉增瑞)

jisuanji bingdu

计算机病毒(computer virus) 计算机系统中一类隐藏在存储介质上蓄意破坏的捣乱程序。它具有可运行性、可复制性、传染性、潜伏性、欺骗性、精巧性、隐藏性和顽固性等特点,对计算机系统与网络的安全和正常运行具有极大的危害性。

计算机病毒的病理机制与人体感染细菌和病毒的病理现象十分相似。它能够通过修改或自我复制向其他程序扩散(传染),进而扰乱系统及用户程序的正常运行。

计算机病毒的破坏过程如图1所示:

开始前病毒程序寄生在介质上的某个程序中,处于静止状态,一旦带病毒程序被引导或调用,它就被激活,变成有感染力的动态病毒,当传染条件满足

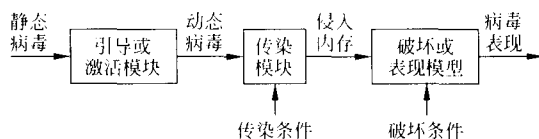


图1 计算机病毒破坏过程图

时,病毒就侵入内存,随着作业进程的发展,它逐步(有时很快地)向其他作业模块扩散,并传染给其他软件。在破坏条件满足时,它就由表现模块或破坏模块把病毒以特定的方式表现出来,干扰系统的正常运行。

病毒可分为以下几种类型:寄生病毒、存储器驻留病毒、引导区病毒、隐形病毒、多形病毒和宏病毒等。

反病毒软件已发展了以下4代:简单的扫描程序、启发式的扫描程序、行为陷阱和全方位的保护。还有两种更为先进的反病毒方法是类属解密技术和数字免疫系统。类属解密技术使得反病毒程序可以容易地检测出复杂的多形病毒,同时保持快速的扫描速度。当包含了一个多形病毒的文件在执行时,病毒必须解密自身来激活。数字免疫系统是一个防止病毒侵犯的综合方法,它的成功依赖于病毒分析机制检测新的病毒血统的能力,通过持续地分析和监视找到最新病毒来连续更新数字免疫系统。

参考文献

- 胡道元主编. 网络设计师教程. 北京:清华大学出版社,2001

(胡道元)

jisuanji cu

计算机簇(computer cluster) 将多台工作站(或微型计算机)用互联网连接起来,统一调度,协调处理,以实现高效并行计算的分布式计算机系统。系统中的计算机可以是同构的,也可以是异构的。如果所用的计算机是工作站,称为**工作站机群**或**工作站网络(NOW)**。

图1是计算机簇的硬件结构图。它由工作站和互联网两部分组成,工作站上增加一个主机接口实现连网。工作站包含高速定点和浮点处理器、大容量存储器、良好的网络支撑环境和图形显示设备以及友好的用户界面。互联网通常使用**局域网**,如**以太网**、**令牌环网**、**光纤分布式数据接口网**等,也可以是**交换网络**,如**异步传送模式(ATM)**、**交换式高速以太网**等。

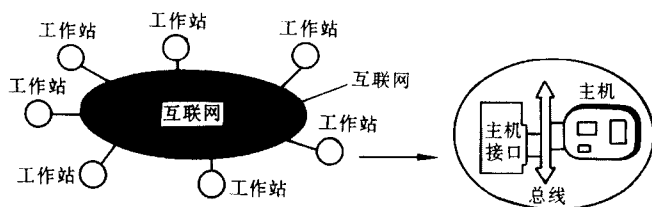


图1 计算机簇的硬件结构

图2是计算机簇的软件结构图。其中操作系统一般为UNIX。为了支持某些特殊的操作,可以对原有的操作系统进行修改和重建。通信协议实现工作站到互联网的数据通信服务。通信原语库由用户编程调用,并行程序设计环境和并行工具包为用户提供方便、友好的使用接口。

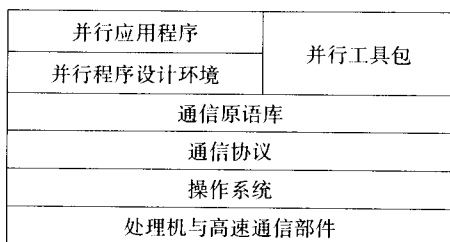


图2 计算机簇的软件结构

具有高性能工作站和互联网以及良好使用环境的计算机簇可以支持多种计算模型。在功能并行中,每台工作站并行执行应用程序的部分功能,而在数据并行中,每台工作站并行处理应用程序的部分数据。在两种并行模型中,互联网都承担工作站间数据交换和信号传输的任务。

计算机簇设计和实现时要解决两个关键问题:一是从各个角度想方设法降低处理机或进程间的通信开销,使系统的资源主要用于计算,从而获得较高的加速比和利用率。采用的方法是针对并行计算的特点,研究和设计高效的精简通信机制和协议,并将底层协议用高速通信部件来支持,大幅度降低通信开销,提高并行效率,从而使系统的性能更高,适用面更广。同时采用快速以太网、光纤分布式数据接口或异步传送模式等技术。二是设计和实现一套友好的、高效的并行程序开发环境和工具系统。用户可以用系统支持的传统FORTRAN, C, C++等语言编程,使用户能继承已有的软件财富。用户还可用系统提供的通信原语库、输出输入操作、并行图形处理、可视化性能评测工具、并行程序调试工具以及人

机交互集成开发工具等并行处理工具包,对程序的并行运行进行控制、调试、监测和评价等工作。

计算机簇具有用户投资风险小,编程方便,系统结构灵活,性能价格比高,可伸缩性好等特点。(郑纬民)

jisuanji daishu

计算机代数 (computer algebra)

研究代数算法的设计、分析、实现及其应用的学科。代数算法有简洁的形式化描述,有基于数学理论的严格证明。代数算法在计算机上用软件实现时,输入和输出都是代数符号表示。计算机代数处理非数值计算。计算机代数又称为符号计算或公式推演。

计算机代数包括各类数学问题的解的算法——从初等数学(如初等代数式化简,初等数论)到高等数学(如解微分方程)各类问题的求解算法,以及这些算法的具体实现和在各学科中的应用。

化简程序 化简就是利用等价关系进行变换,将集合 S 上的元素变换为 S 上的另一个“简单”的元素。设 T 是变换, \sim 是集合 S 上的等价关系, $u \in S$,则这个化简可用

$$T(u) \sim u \quad \text{和} \quad T(u) \leq u$$

来描述,其中 \leq 是描述“简单”性的偏序。在不同场合,它有不同的含义。例如,它可以是“ $T(u)$ 比 u 的字符少”,“求 $T(u)$ 的值比求 u 的值容易”,“ $T(u)$ 比 u 更容易理解”,“ $T(u)$ 比 u 更便于下一步使用”等等。变换 T 称为化简器。两个恒等的表达式经规范化简器化简后所得形式可能不一样,即可以是 S 内不同的元素。为了使得两个恒等的表达式经化简后变为同一形式,即 S 的同一元素,应使用规范化简器 T ,它可描述如下:对任一 $u, u \in S$

$$T(u) \sim u, \quad T(u) \leq u \\ u \sim v \Leftrightarrow T(u) = T(v)$$

这样, S 内所有与 u 等价的元素有唯一的元素 $T(u)$ 与之对应。 $T(u)$ 代表了一个等价类,称为 u 的规范形式。

代数扩域法 近世代数的理论已经证明: F 的代数扩域 $F(\alpha)$ 与多项式环 $F[\alpha]$ 是同构的,而且代数扩域 $F(\alpha)$ 的元素都可用多项式环 $F[\alpha]$ 的元素

$$a_n \alpha^n + a_{n-1} \alpha^{n-1} + \cdots + a_0$$

来表示,且这种表示的约简形式是惟一的。同样代数扩域 $F(\alpha_1, \alpha_2, \cdots, \alpha_m)$ 的元素也可以用多项式环

$F[\alpha_1, \dots, \alpha_m]$ 的元素表示。欲化简一个表达式 S , 先构造一个代数扩域 E , 使得 $S \in E$; 然后在 E 上化简 S , 这就是代数扩域法。

多项式无平方分解 如果 $p(x)$ 没有重根, 则称为无平方多项式。设 $p(x) \in R[x]$, $p(x)$ 的无平方分解是

$$p(x) = p_1(x)p_2^2(x) \cdots p_k^k(x)$$

其中, $p_k(x) \nmid p_i(x)$ 都是无平方多项式, 且当 $i \neq j$ 时, $p_i(x)$ 和 $p_j(x)$ 互素。推演无平方分解式的算法称为无平方分解算法。

Berlekamp 算法 在 Z_p (其中 p 是素数) 上将一元多项式因式分解的算法, 其时间复杂度是 $O(n^3 + prn^2)$, 其中 r 是因子数, n 是多项式的次数。

中国剩余定理 设 p_0, p_1, \dots, p_{n-1} 是两两互素自然数, $p = p_0 \times p_1 \times \cdots \times p_{n-1}$, 则任一小于 p 的非负整数 u , $0 \leq u < p$, 可用惟一组剩余 u_0, u_1, \dots, u_{n-1} 表示, 其中

$$u_i \equiv u \pmod{p_i}, \quad i = 0, 1, \dots, n-1$$

给定 p_0, p_1, \dots, p_{n-1} 时, 可记作

$$u \leftrightarrow (u_0, u_1, \dots, u_{n-1})$$

反之, 若已知 $(u_0, u_1, \dots, u_{n-1})$ 且 $0 \leq u_i < p_i$, $i = 0, 1, \dots, n-1$, 则

$$u = \sum_{i=0}^{n-1} c_i d_i u_i \pmod{p}$$

其中 $c_i = p/p_i$, $d_i c_i \equiv 1 \pmod{p_i}$ 。研究求 u 的快速算法是计算机代数的重要课题。

符号微分 函数的导函数和偏导函数称为符号导函数, 但习惯上常称为符号微分。根据数学公式可设计计算符号微分的递归算法。

符号积分 函数的不定积分、参变积分称为符号积分。符号积分是一个比符号微分困难得多的问题。困难在于: 有的函数, 如 e^{x^2} , 原函数存在, 但又从理论上证明了它没有闭形式解, 通俗地说就是“积不出来”。

计算机代数语言 比较有影响的有 ALTRAN, CAMAL, FORMAL, MACSYMA, muMATH, REDUCE, SAC-2, SCRATCHPAD, MATHEMATICA。我国自己研制的有 CASC 等。

计算机代数有着广泛的应用, 应用范围从纯数学到工业部门。例如, 它已用于数论、射影几何、特殊函数、微分方程、天体力学、相对论、高能物理、理论物理、等离子体物理、电动力学、流体力学、理论化学、教育等学科。也已用于船体设计、直升飞机设计、电子显微镜的设计、机器人的手臂运动、铀采矿

工厂的设计等等工业部门。

参考文献

1. Buchberger B, et al. Computer Algebra: Symbolic and Algebraic Computation. New York: Springer-Verlag, 1983
2. 衷仁保. 计算机代数. 长沙: 国防科技大学出版社, 1989
3. 衷仁保. 符号计算语言 CASC 及编译实现. 南昌: 江西高校出版社, 1993 (衷仁保)

jisuanji dianyuan

计算机电源 (power supply for computer)

为计算机主机和外围设备供电的设备。大中型计算机对电源除了容量和质量的要求外, 还有系统控制、检测和保护等要求, 即专门设有供配电和监控装置, 把众多的交、直流电源联成一体, 以保证计算机安全可靠地运行。通常在交流电网的输入端安装不间断电源 (UPS), 在正常工作状态下, 它对输出的交流起稳定电压、稳定频率和抗干扰的作用; 在交流电网发生故障时, 它能向负载继续供电一段时间。大中型计算机直流供电有两种方式: 一种是将交流电网的交流电变换成 50V 直流总线电压, 然后用直流-直流 (DC-DC) 模块将总线电压变换成所需要的直流电压。直流总线和蓄电池相并联, 当电网发生故障时, 由蓄电池供电。另一种方式是把交流电压送到计算机的各个部分, 再就地转换成所需要的直流电压。

直流电源有磁放大器电源、整流电源、线性电源、开关电源等, 最常见的是线性电源和开关电源。开关电源又可分为串联式开关电源和无工频变压器开关电源两种。上述三种常见的稳压电源的性能比较见表 1。

三种电源都用功率管作调整管, 多数线性电源中的功率三极管工作在线性区, 而开关电源中功率三极管开通时工作在饱和区, 截止时进入截止区, 其功耗要小得多, 有利于缩小体积。在市电供电的条件下, 线性电源和串联式等基本形式的开关电源往往要用工频 (50Hz, 电网频率) 变压器, 而无工频变压器开关电源 (一般有高频变压器) 省去了笨重的工频变压器, 因此, 它是微型计算机普遍采用的。从表 1 可见, 无工频变压器的开关电源尺寸小, 重量轻, 效率高。此外, 无工频变压器开关电源一般比线性电源具有更多的内部储存能量, 有利于在电网瞬时中断时保持工作。但线性电源的稳定度高, 输出纹波小, 常用于一些有特殊要求的外围设备中。

表1 三种稳压电源性能比较

比较项目 \ 电源类型	线性稳压电源	串联式开关稳压电源	无工频变压器 开关稳压电源
电压稳定度(电网变化 $\pm 10\%$)	0.01% ~ 0.05%	0.1% ~ 0.2%	0.1% ~ 0.2%
负载稳定度(负载变化 0 ~ 100%)	0.02% ~ 0.05%	0.1% ~ 0.2%	0.1% ~ 0.2%
输出纹波(峰-峰值)	< 5 mV	10 ~ 100 mV	10 ~ 100 mV
上升建立时间	100 μ s	100 ms	100 ms
电压维持时间	5 ms	25 ms	25 ms
过渡过程	0.02 ~ 0.1 ms	0.5 ms 至数 ms	0.5 ms 至数 ms
效率	30% ~ 40%	40% ~ 70%	60% ~ 85%
体积	1	2/3 ~ 1/3	1/3 ~ 1/5
重量	1	2/3	1/3
可靠性(MTBF)	100 000 h	50 000 ~ 100 000 h	30 000 ~ 100 000 h
控制电路元件个数	10 ~ 30 个	10 ~ 50 个	30 ~ 120 个

随着计算机容量、速度和组装密度的不断提高,对电源装置的性能指标,特别是电源的尺寸、重量、可靠性以及可维性等要求逐步提高。它促使电源朝高功率密度和高可靠的方向发展。

开关电源高频化是电源小型化的重要途径,常用开关电源的工作频率为 20 kHz ~ 50 kHz,还将提高到 100 kHz ~ 500 kHz 以上。频率的提高不仅使电源重量减轻,体积减小,效率提高,动态特性和小信号扰动下的稳定性改善,并且使传导或辐射的电磁干扰(EMI)容易被滤波、屏蔽和抑制。(方资端)

jisuanji donghua

计算机动画 (computer animation) 用计算机生成一系列可供实时演播的画面的技术。它可辅助传统卡通动画片的制作,也可通过对三维空间中虚拟摄像机、光源及物体运动和变化(形状、色彩等)的描述,逼真地模拟客观世界中真实的或虚构的三维场景随时间而演变的过程。所生成的一系列画面可在显示屏上动态演示,也可将它们记录在电影胶片上或转换成视频信息输出到录像带上。

计算机动画的研究始于 20 世纪 60 年代初,1963 年美国 AT&T Bell 实验室制作了第一部计算机动画片。在 80 年代之前,主要集中于二维动画系统的研制,应用于教学演示和辅助传统的动画片制作。三维计算机动画的研究始于 70 年代初,至 80 年代中后期,由于具有实时处理能力的超级图形工作站的出现,三维几何造型技术和真实感图形生成技术取得很大进展,才促进了具有高度逼真效果的三维计算机动画技术迅速发展,并达到实用化、商品化的地步。到 90 年代,将计算机动画技术应用于电影特技取得了显著成就。在《魔鬼终结者》、《侏罗纪公

园》、《玩具总动员》和《泰坦尼克号》等电影中,人们可以充分领略计算机动画的高超魅力。

动画的原理是利用人的视觉残留特性使连续播放的静态画面相互衔接而形成动态效果。计算机制作动画的过程是用绘制程序生成一系列的景物画面,其中当前帧画面是对前一帧画面的部分修改。动画是运动中的艺术,正如动画大师 John Halas 所说,运动是动画的要素。一般来说,计算机动画中的运动包括:①景物位置、方向、大小和形状的变化;②虚拟摄像机的运动;③景物表面纹理、色彩的变化。计算机动画所生成的是一个虚拟的世界,虚拟景物可以是商标、汽车、建筑物、人体、分子、桥梁、云彩、山脉、恐龙或昆虫等,虚拟景物并不需要真正去建造,物体、虚拟摄像机的运动也不会受到什么限制,动画师可以随心所欲地创造他的虚幻世界。

计算机动画的制作主要包含如下步骤:①创意 根据设计的需要,由导演设计好动画制作的脚本;②预处理 扫描外部图像,输入外部资料;③场景造型;④设定材质和光源;⑤设置动画;⑥运动图像的绘制;⑦动画播放;⑧后处理;⑨动画的录制;⑩配音(包括背景音乐和台词)。

后处理在计算机动画中占据非常重要的位置。后处理是指动画后期的非线性编辑合成技术,包括蓝屏抠像、合成、图像形态变换、特殊光效等。理论上,它不属于计算机动画的范畴,但它是动画制作中必不可少的过程。特别是视频技术的数字化趋势,使该技术日益受到人们的重视。目前,已有许多优秀的后期合成软件。在这些软件中,大量采用了图像处理和计算机视觉技术(如摄像机反求),可方便地进行特技效果的处理和运动跟踪、深度合成等复杂操作。

计算机动画主要研究运动控制技术以及与动画有关的造型、绘制、合成等技术。例如：①动画形体造型技术；②动画运动控制和描述；③动画图像绘制技术和算法；④动态模拟、动画系统的集成环境；⑤关节体、人体动画；⑥动画语言与系统；⑦用于动画运动控制和生成的专门硬件设备及接口；⑧特殊视觉效果生成技术。

动画技术有很多,要对它们进行细致的分类是困难的,大致可分为关键帧动画、渐变和变形物体动画、过程动画、关节动画和人体动画、基于物理的动画等。计算机动画技术的成熟推动了动画软件的发展,目前有大量优秀的商用动画软件可供选用。

计算机动画可广泛应用于电影特技、商业广告、电视片头、动画片、游戏、网页设计、计算机辅助教育、军事演习、建筑设计、飞行模拟等。

参考文献

1. 鲍虎军,金小刚,彭群生等. 计算机动画的算法基础. 杭州:浙江大学出版社,2000

2. 齐东旭,马华东,黄心渊等. 计算机动画原理与应用. 北京:科学出版社,1998

(金小刚 王裕国)

jisuanji fangzhen

计算机仿真 (computer simulation) 利用计算机建立、校验、运行实际系统的模型以得到模型的行为特性,从而达到分析、研究该实际系统之目的过程、方法和技术。这里的“系统”是广义的,它包括工程系统,如电气系统、热力系统、计算机系统,也包括非工程系统,如交通管理系统、生态系统、经济系统等。

现代仿真技术均是在计算机支持下进行的,因此,系统仿真也称为计算机仿真。计算机仿真有三个基本的活动,即系统建模、仿真建模和仿真实验,联系这三个活动的是计算机仿真的三要素,即系统、模型、计算机(包括硬件和软件)。它们的关系可用图1描述。

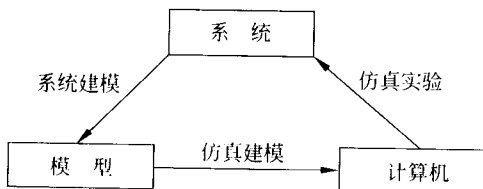


图1 计算机仿真三要素及三个基本活动

计算机仿真的类型

可以从不同的角度对计算机仿真加以分类。比较典型的分类方法是:根据仿真所采用的计算机类型分类、根据仿真时钟与实时时钟的比例关系分类及根据系统模型的特性分类。

根据仿真计算机类型分类 自从数字计算机出现以后,其高速计算能力和巨大的存储能力使得复杂的数值计算成为可能,数字仿真技术得到蓬勃的发展,从而使仿真形成一门专门学科——系统仿真学科。

按所使用的仿真计算机类型也可将仿真分为三类:模拟计算机仿真,数字计算机仿真和数字模拟混合仿真。

模拟计算机本质上是一种通用的电气装置,这是20世纪50年代至60年代普遍采用的仿真设备。将系统数学模型在模拟机上加以实现并进行实验称为模拟机仿真。

数字计算机仿真是将系统数学模型用计算机程序加以实现,通过运行程序来得到数学模型的解,从而达到系统仿真的目的。

20世纪60年代至70年代,在数字计算机技术还处于较低水平时,产生了数字模拟混合仿真,即将系统模型分为两部分,其中一部分放在模拟计算机上运行,另一部分放在数字计算机上运行,两个计算机之间利用模/数和数/模转换装置交换信息。

随着数字计算机技术的发展,其计算速度和并行处理能力的提高,模拟计算机仿真和数字模拟混合仿真已逐步被全数字仿真取代,因此,今天的计算机仿真一般指的就是数字计算机仿真。

根据仿真时钟与实际时钟的比例关系分类 实际动态系统的时间基称为实际时钟,而系统仿真时模型所采用的时钟称为仿真时钟。根据仿真时钟与实际时钟的比例关系,系统仿真分类为:

(1) 实时仿真 即仿真时钟与实际时钟完全一致,也就是模型仿真的速度与实际系统运行的速度相同。

(2) 亚实时仿真 即仿真时钟慢于实际时钟,也就是模型仿真的速度慢于实际系统运行的速度。有时也称为离线仿真。

(3) 超实时仿真 即仿真时钟快于实际时钟,也就是模型仿真的速度快于实际系统运行的速度。参见连续系统仿真。

根据系统模型的特性分类 仿真基于模型,模型的特性直接影响着仿真的实现。Orén进行了总结,将模型形式加以分类如表1所示。

表1 模型分类

模型描述变量的轨迹	模型的时间集合	模型形式	变量范围	
			连续	离散
空间连续变化模型 空间不连续变化模型	连续时间模型	偏微分方程	√	
		常微分方程	√	
离散(变化)模型	离散时间模型	差分方程	√	√
		有限状态机		√
		马尔可夫链		√
	连续时间模型	活动扫描	√	√
		事件调度	√	√
		进程交互	√	√

从仿真实现的角度来看,系统模型特性可分为两大类:一类称为连续系统,另一类称为离散事件系统。由于这两类系统固有运动规律的不同,因而描述其运动规律的模型形式就有很大的差别。相应地,系统仿真技术也分为两大类:连续系统仿真和离散事件系统仿真。

计算机仿真的一般步骤

计算机仿真的一般步骤可用图2来描述。

(1) 针对实际系统建立其模型 建模与形式化的任务是:根据研究和分析的目的,确定模型的边界并对模型进行形式化处理,以得到计算机仿真所要求的数学描述。

(2) 仿真建模 其主要任务是:根据系统的特点和仿真的要求选择合适的算法,当采用该算法建立仿真模型时,其计算的稳定性、计算精度、计算速度应能满足仿真的需要。

(3) 程序设计 即将仿真模型用计算机能执行的程序来描述。

(4) 程序检验 其目的一方面是程序调试,更重要的是要检验所选仿真算法的合理性。

(5) 进行模型实验 这是实实在在的仿真活动。

(6) 仿真输出分析 这既是对模型行为数据的处理,以便对系统性能作出评价,同时也是对模型的可信性进行检验。

在实际的仿真时,上述每一个步骤往往需要多次反复和迭代。

仿真技术的应用

仿真技术作为一门独立的学科已经有50多年的发展历史,它不仅用于航天、航空、各种武器系统

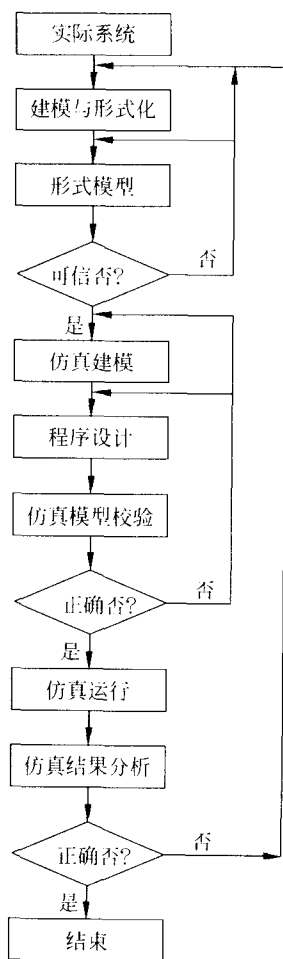


图2 计算机仿真的一般步骤

的研制部门,而且已经广泛应用于电力、交通运输、通信、化工、核能等各个领域。特别是,近20年来,随着系统工程与科学的迅速发展,仿真技术已从传统的工程领域扩展到非工程领域,因而在社会经济系统、环境生态系统、能源系统、生物医学系统、教育训练系统也得到了广泛的应用。

例如,在教育训练系统,凡是需要有一个或一组熟练人员进行操作、控制、管理与决策的实际系统,都需要对这些人员进行训练、教育与培养。为了解决这个问题,各类**训练仿真器**被开发出来。它能模拟实际系统的工作状况和运行环境,又可避免采用实际系统培训时可能带来的危险性及高昂的代价(参见**训练仿真器**)。

又如,在产品的设计、开发、制造的各个阶段,仿真技术从过去单项发展到多学科的协同,最具代表性的是20世纪90年代产生的虚拟制造技术。

虚拟制造中的“制造”系指“大制造”,即包括产品设计、开发、加工、生产全过程。是各种**计算机辅助技术**面向产品全生命周期的集成化综合应用,涉及的技术领域颇为广泛,包括**计算机辅助技术**、建模与仿真技术、**虚拟现实技术**、产品开发管理技术等。而建模与仿真技术是虚拟制造的核心技术。虚拟制造采用计算机建模与仿真技术,以数字模型为基础,采用高逼真度仿真手段,以增强制造过程各个阶段的决策与控制能力。

总之,仿真技术之所以得到迅速发展,其根本的动力来自应用。仿真技术正是从其广泛的应用中获得了日益强大的生命力,而仿真技术的发展反过来使其得到愈来愈广泛的应用。

参考文献

1. 肖田元,张燕云,陈加栋. 系统仿真导论. 北京:清华大学出版社,2000
2. Wiens G J. An Overview of Virtual Manufacturing [A]. Virtual Manufacturing-Proceedings of the 2nd Agile Manufacturing Conference (AMC'95) [C]. Albuquerque, New Mexico, USA 1995, ERI Press. 233 ~ 243
3. 肖田元,韩向利等. 虚拟制造的定义与关键技术 [J]. 清华大学学报(自然科学版), 1998, 10 (肖田元)

jisuanji fuzhu gongyi guihua

计算机辅助工艺规划 (computer aided process planning, CAPP) 利用计算机自动设

计产品零件机械加工工艺规程的技术。机械加工工艺规程设计所要解决的主要问题是根据产品零件的几何形状、尺寸、加工精度、表面粗糙度和热处理等技术要求,确定产品零件的加工方法,加工顺序,所需的机床、刀具、夹具、量具和切削参数等。综合应用**计算机图形学**、**工程数据库**和**人工智能**等计算机技术解决这一复杂的多目标规划问题,不但克服了手工设计工艺规程存在的效率低、一致性差、难以积累和充分利用工艺专家的经验及知识等缺点,而且提高了工艺规程的质量和标准化规范化程度,可对工艺技术文件进行统一的管理,达到缩短产品生产准备周期、降低生产成本从而增强企业市场竞争能力的目标。CAPP是连接计算机辅助设计(CAD)和计算机辅助制造(CAM)的桥梁,已成为实现计算机集成制造系统(CIMS)的关键环节之一。

计算机辅助工艺规划(CAPP)的研究可追溯到20世纪60年代末。CAPP发展史上的一个重要里程碑是美国J. Jikoff等人于1976年开发的自动工艺规划系统(CAMI)。此后,CAPP的研究已成为制造自动化领域的重要课题。

CAPP的实现主要归纳为派生式和创成式两类。目前实用的CAPP系统主要采用派生式。

派生式CAPP系统的基本原理是利用零件的相似性,即相似零件有相似的工艺规程。一个新零件的工艺规程是通过检索系统中已有的相似零件的工艺规程并加以自动筛选或编辑而成。根据零件信息的描述与输入方法不同,派生式CAPP系统又分为基于成组技术(GT)与基于特征的派生式系统。基于GT的派生式CAPP系统的工作过程如下:

- (1) 选择零件分类编码系统,对现有加工零件按编码法则进行编码,以工艺相似性将零件分组(簇),并为每组的代表零件(样件)制定标准工艺规程。
- (2) 根据零件分类编码系统,给待编工艺规程的新零件编码。
- (3) 根据新零件的编码,查找所属的零件族,找到后调出该零件组的标准工艺规程,并根据零件信息,对标准工艺规程进行编辑加工,生成新零件的加工规程。

创成式则利用计算机的特点,模拟工艺员对零件进行各种逻辑判断、决策和计算,自动生成工艺规程。创成式CAPP系统的工作过程如下:

- (1) 零件特征信息建模 产品零件是具有特定几何形状、工艺性质的物体。在制造过程中,零件的

形状、尺寸公差和技术要求是决定工艺规程的主要因素。因此完整、准确、有效地抽取和表达零件信息是实现 CAPP 系统的一个关键问题。目前建立零件信息模型的方法主要有如下 3 种:第一种是通过人机交互直接获取 CAPP 所需的各类数据和信息;第二种是从 CAD 建立的实体模型中抽取特征,建立 CAPP 的零件信息模型;第三种是建立统一的产品零件的特征模型,通过 STEP 标准或数据库实现与 CAD、CAM 系统的数据转换。

(2) 建立工艺数据库 工艺数据库是自动工艺规划所必需的基础数据,是关于加工能力和工艺手册数据的集合。它包括加工方法库、机床库、刀具库、夹具库、量具库和切削参数库等。

(3) 进行自动工艺规划 包括零件工艺过程主干生成、工序设计。

(4) 进行工艺文档管理 其主要功能有工艺卡片输出、工序图绘制、工艺文件交互编辑及事务性管理(如登记、审核等)。

进入 20 世纪 80 年代后,以专家系统技术为基础的创成式 CAPP 系统受到了广泛地重视。这类 CAPP 系统具有模块化的优点,知识库中的工艺知识易修改和扩充,适合在不同制造环境下应用。专家系统通常由知识库和推理机两部分组成。知识库包括两大类知识:事实性知识和启发性知识。有经验的工艺师求解工艺规划问题的经验被合理地组织在知识库中。

CAPP 专家系统的推理机控制系统的运行,是利用系统所具有的工艺知识,按照一定策略进行自动工艺规划。CAPP 知识库与推理机的分离使得知识库和推理机的修改可以互相独立,互不影响,尤其适合 CAPP 系统依赖于制造环境这一特点。同时 CAPP 专家系统也具有良好的人机交互界面和解释机制,并提供推理失败时的处理功能。

我国的 CAPP 研究工作始于 20 世纪 70 年代末期,1986 年国家的高技术发展计划(亦称 863 计划)正式将 CAPP 列为计算机集成制造系统(CIMS)主题的一个研究专题,并取得若干应用成果。

当前 CAPP 技术和系统的主要发展趋势是集成化、智能化和工具化。

参考文献

1. Chang T C. Expert Processing Planning for Manufacturing. Addison Wesley Publishing Company, 1990
2. Pekline J, Sluge A. Contribution to Develop-

ment of a Generative CAPP System Based on Manufacturing Process Topology. Analysis of the CIRP, 1989, (38)

3. CIMS 主题专家组. 计算机集成制造技术与系统的发展趋势. 北京: 科学出版社, 1994

4. 戴同. CAD/CAPP/CAM 基本教程. 北京: 机械工业出版社, 1996 (孔繁胜 林兰芬)

jisuanji fuzhu huitu

计算机辅助绘图 (computer aided drafting)

利用计算机及图形设备进行图形的生成、处理和绘制的过程和技术。这里所说的图形主要包括二维零件图、逻辑图、电路图和三维线框图等工程图。广义地也包含实体图、广告图、美术图、艺术图等多种类型。

计算机辅助绘图与计算机辅助设计(CAD)技术联系紧密,它是 CAD 的一个重要组成部分。在 CAD 系统中,自动绘图贯穿于计算机辅助设计的各个阶段,如绘制二维工程图、三维线框图、各类视图和立体图、分析计算结果图、加工仿真图等。CAD 系统中的绘图模块具有与三维造型软件的接口,它能把三维造型软件构造的实体,按工程需要产生各种视图,并通过输出设备绘制这些视图的工程图纸。

在许多应用领域中,往往需要将已有的工程图纸转换为电子图纸,以使用计算机进行编辑修改。为此应通过扫描仪将工程图输入计算机,使其表示为光栅图像格式,然后进一步采用图文分离、图形与字符识别等技术将光栅图像转换为矢量图形格式。这项技术称为工程图纸向量化。

我国于 20 世纪 60 年代后期开始计算机辅助绘图的研究,先后研制成功平台式小型绘图仪、大型绘图仪,并开发出一批计算机辅助绘图软件。目前,计算机辅助绘图已经广泛应用于航空航天、船舶、汽车、土木建筑、机械、电子、轻工、化工、纺织等行业以及气象、地质等部门。

参考文献

1. Foley J D, van Dam A, Feiner S K, Hughes J F. Introductions to: Computer Graphics. Addison Wesley, 1993
2. 孙家广, 杨长贵. 计算机图形学(新版). 北京: 清华大学出版社, 1995 (陈德人)

jisuanji fuzhu jishu

计算机辅助技术 (technology for computer

aided something) 采用计算机作为工具,辅助人在特定应用领域内完成任务的理论、方法和技术。它包括诸如计算机辅助设计(CAD)、计算机辅助制造(CAM)、计算机辅助教学(CAI)等领域。“辅助”是强调了人的主导作用,计算机和使用者构成了一个人机密切交互的系统。

随着计算机科学技术的飞速发展,计算机辅助技术的应用领域不断拓广,应用水平不断提高。CAD和CAM首先在飞机、汽车和船舶等大型制造业应用中趋于成熟,发展了许多共性的技术,开发出许多公用的工具软件和应用软件,其应用逐步推广到机械、电子、轻纺和服装等产品的制造业以及建筑、土建等工程项目。同时,它的技术和方法也被推广到许多新的计算机辅助领域,例如计算机辅助工艺规划(CAPP)、计算机辅助测试(CAT)、计算机辅助质量控制(CAQ)以及用于教学和培训目的的计算机辅助教学(CAI)等。

1962年麻省理工学院I. E. Sutherland的博士论文“Sketchpad:一个人机通信的图形系统”,第一次提出设计者可以用光笔和显示屏幕作为媒介以图形方式和计算机进行交互,这就是最初的交互式系统。但Sketchpad的第一版本只限于画二维图,1963年由T. E. Johnson开发的更新版本可以建立对象的三维线框模型,因此可以显示出从不同角度观察的透视图,据此可画出工程三视图。Sketchpad实现了CAD的初步目标:利用计算机辅助设计和绘制工程图纸,它所发展的计算机图形学和人机交互技术,不但是所有计算机辅助系统的基础,而且也是计算机应用技术中迄今仍然非常活跃的两个重要分支。

因为在各种计算机辅助系统的运作过程中都会频繁地涉及到大量信息和数据,因此数据库管理系统也是它们的重要组成部分。但是目前已较为成熟的商用数据库管理系统常常不能满足其需要。例如CAI系统中处理的常是多种媒体信息,因此需要多媒体数据库。在CAD或CAM环境下提供服务的工程数据库,其概念模式要描述成千上万个不同类型的实体,相互间关联非常复杂,而且不稳定,需经常修改。这些新型数据库管理系统正处于技术上不断完善并进入实用化的阶段。

计算机辅助设计(CAD) 一个设计过程可以简要地归结为:首先将设计目标形式化为规约。设计师根据规约并运用他的知识,包括基本原理、经验知识和对同类产品已有的了解,建立设计对象的概念模型。由此设计模型再对照规约进行分析和评估,

找出与规约的偏差,再返回修改概念模型,如此反复循环,直至得到满意的或者最优的设计。设计进程结束输出机械图、逻辑图、电路图和其他设计方案的描述文档。CAD各项技术对于设计全过程的支持是明显和有效的,表现在:①**计算机图形学、人机交互技术和合适的数据库**将使模型的建立、修改和存储方便、高效。②对于不同的设计领域,例如机械设计要求计算结构或零件的强度,要仿真机构的几何运动;电子设计需要分析电路特性和逻辑功能等,都有大量的针对性应用软件可在计算、分析、评估中选用。③通过计算机网络可以共享设计数据,实现异地设计者的协同设计。由此使CAD显著提高了设计效率和质量。

计算机辅助制造(CAM) CAM是通过计算机与生产设备的直接或间接联系,对制造工厂的作业进行设计、管理和控制的过程与技术。CAM的核心是计算机数值控制(CNC)。有关计算机辅助制造的内容可参见条目**计算机辅助制造**。

计算机辅助工艺规划(CAPP) 在工件图纸设计完成提交数控机床加工之前,有时还要经过工艺过程规划。即决定从原材料到工件成品之间加工顺序和所需机床、刀具等。这项工作需要有丰富生产经验的工程师进行复杂的规划。计算机辅助工艺规划(CAPP)也是运用人机交互、计算机图形学、工程数据库以及专家系统等计算机科学技术来实现的。CAPP常是联结CAD与CAM的桥梁。在集成化的CAD/CAPP/CAM系统中,由于设计时在公共数据库中所建立的产品模型不仅包含了几何数据,也记录了有关工艺需要的数据,以供CAPP采用。CAPP的设计结果也存回公共数据库中供CAM的数控编程。集成化的作用不仅在于免去了人工传递数据,更有利于产品生产的整体考虑。从公共数据库,使设计工程师可以获得并考察其所设计产品的加工信息;制造工程师可以从中清楚地知道产品的设计需求。全面地考察这些信息,可以使产品生产获得更大的效益。

计算机集成制造系统(CIMS) 在产品生命周期中各项作业所采用的计算机辅助系统,如CAD、CAPP、CAM、CAT、CAQ等,原来都是制造过程中的“自动化孤岛”,追求每一单个过程的优化不一定能够达到企业整体优化的目标——缩短产品设计时间,降低产品的成本和价格,改善产品的质量和服务质量以提高产品的市场竞争力。CIMS就是将各个单项的计算机辅助技术以及制造信息系统(如ERP

等)集成在一起,从而使全局优化的效果更为明显。

计算机辅助教学(CAI) 计算机辅助技术另一个重要的应用领域是计算机辅助教学(CAI)。在CAI中,更为广泛地采用多种媒体信息——文本、声音、图形、图像、影视等进行人与计算机的信息交流,并引入人工智能技术,创造了良好的学习环境,有助于更好地调动学生的学习热情和主动性,启发思考,加深理解,促进教育普及和获得好的教育效果。正在迅速发展的网络教学可以看作是CAI在网络上的延伸,网上用户可以共享CAI提供的各种教学资源。(何志均 董逸生)

jisuanji fuzhu jiaoxue

计算机辅助教学(computer assisted instruction, CAI) 为弥补传统教学模式的不足和提高教学质量与教学效率,利用计算机辅助教学活动的过程、方法与技术。

自20世纪80年代以来,计算机在教育领域中的应用得到了迅速发展,其中特别受到重视的是计算机辅助教学和计算机管理教学(CMI),后者指的是利用计算机管理教学,如学生注册、选课、排课、教学评估、学分统计、成绩记录与查询等。CAI和CMI是计算机辅助教育(CBE)的两个主要组成部分。

计算机辅助教学(CAI)综合应用了多媒体、计算机网络、人工智能、知识处理等多种技术,它既可作为常规课堂教学的补充手段,也可以部分地替代教师进行课程的教学。对于那些用语言文字难以表达的抽象内容、动态的变化过程和复杂的微观结构等,CAI通过使用图表、动画、声音等多媒体技术进行演示或模拟其变化过程,改善教学效果。采用CAI技术的训练课和习题课,以学生为中心,按各自的进度进行,适合个性化教学,能充分发挥学生的学习主动性。CAI利用计算机的交互特性实现双向教学,克服传统教学中学生只是单向接受知识的被动局面。

计算机辅助教学(CAI)的教学模式有多种。例如,“模拟实验型CAI”适用于电子电路实验、化学反应实验等,它的教学效果较好,能节省实验费用;“示教型CAI”适用于演示一些抽象的数学、物理现象和过程,是教师的好帮手;“练习自测型CAI”可提高外语、医学、程序设计等课程的课外学习效率,很受学生欢迎;“授课型CAI”有利于自学某些课程,特别适合于职业培训和成人教育,在家教辅导中也能发挥很大作用。

CAI除了必要的硬件和软件之外,课件是最重要的一个部分。通过运行课件,计算机能实现预定的教学计划、过程、方法和策略,支持教学活动的全过程。课件的水平和质量对教学效果起着决定性的作用。制作CAI课件既要按照选题确定相应的教学模式(例如操练型、辅导型、咨询型、模拟型、游戏型等),还要恰如其分地运用多媒体技术的表现手段和计算机交互技术,更要在学习理论的指导下,按照人的认知规律指导课件的设计。

在CAI的发展过程中,学习理论对CAI的应用及课件的开发起着重要作用。从20世纪60年代到70年代末,多数CAI软件采用行为主义的学习理论,强调刺激、反应和强化,这一时期开发的CAI课件多数是操练型和个别指导型。80年代的10年,认知学习理论成为指导CAI发展的重要理论基础,它把学习看作是学习者根据自己的态度、需要、兴趣和爱好,利用自己原有的认知结构,对外界刺激所提供的信息主动作出的有选择的信息加工过程。根据认知学习理论,课件开始时利用体现学习内容且具有强烈感染力的画面和声音唤起学生的注意,并告诉学生学习的目标,激起学生对学习的欲望,接下来会通过测试,刺激学生回忆以前的学习内容,以便把已有的与将要学习的新知识结合起来。然后,向学生呈现刺激材料,即呈现教学信息并不断地提供学习指导。为了检验学生对新知识的理解,还提供足够多的练习,让学生“参与”学习,并形成新的刺激,诱发学生的行为。这一时期的CAI软件利用多媒体技术提供许多实例和情境,让学生去求解、去探索,对于发展学生的认知策略也是不可缺少的。

20世纪90年代以来,CAI又进入到一个新的发展阶段,主要表现在建构主义学习理论的应用和计算机网络的使用。建构主义学习理论突出了学习者的主体作用,它认为学习者是信息加工的主体,是意义的主动建构者,“情境”、“协作”、“会话”和“意义建构”是学习环境中的四大要素,教师要由知识的传授者、灌输者转变为学生主动建构意义的帮助者、促进者。

Internet 为开发能体现建构主义理论的学习环境提供了良好的条件,计算机辅助教学开始进入到一个以学生为中心、依托于Internet的全新的阶段。借助Internet的网络教学可以开展多种不同形式的教学活动。例如,教师实时(利用视频会议系统)或非实时(如利用网络课件、视频点播)地向学生讲授课程内容;实时(如利用在线交谈)或非实时(如利

用电子邮件)地对学生进行个别辅导;学生与学生、学生与教师间可利用电子公告板、在线交谈、电子邮件、邮递清单(mailing list)等手段实时或非实时地进行讨论。

以信息技术为代表的科学技术的迅猛发展,使传统以CAI为中心的计算机辅助教育的内涵发生了革命性的变化。20世纪90年代,一些发达国家提出了教育信息化的概念,它是以现代信息技术为基础的新教育体系,包括教育观念、教育组织、教育内容、教育模式、教育技术、教育评价、教育环境等一系列的改革和变化。例如美国教育部在2000年底提出了数字化学习(E-learning)的理念,其核心是通过因特网和其他数字内容进行学习与教学活动,充分利用现代信息技术所提供的、具有全新沟通机制与丰富资源的学习环境,实现一种全新的学习方式,这种学习方式将改变传统教学中教师的作用和师生关系,从而根本改变教学结构和教育本质。

我国在教育信息化方面也开展了大量的工作。1993年启动建设了中国教育与科研网(CERNET);1997年国务院批转教育部《面向21世纪教育振兴行动计划》中提出了“通过现代远程教育工程,形成开放式教育网络,构建全民终身学习体系”的战略思想;2000年10月启动了面向全国基础教育的“校校通”工程;近年所开展的教育政务信息化工程加快了教育系统办公自动化的建设,所有这些都为实现我国教育信息化打下了基础。

参考文献

1. Kulik J A, Bangert Drowns R L. Computer Assisted Learning. Handbook of Educational Ideas and Practices. London: Routledge, 1990
2. 何克抗. CAI的理论基础和以学为中心的课件设计. 中国基础教育网(<http://www.cbe21.com/cai/jiaoyujsh/zongshuwzh/0005.htm>)
3. 何克抗. 计算机辅助教育. 北京: 高等教育出版社, 1997 (张福炎)

jisuanji fuzhu ruanjian gongcheng

计算机辅助软件工程 (computer aided software engineering, CASE) 参见软件工程。

jisuanji fuzhu sheji

计算机辅助设计 (computer aided design, CAD) 采取系统化工程方法, 利用计算机帮助设

计人员完成设计任务的理论、方法和技术。它综合了计算机图形学、人机交互技术、工程数据库和设计方法学等多个领域的理论、方法和技术, 建立具有辅助设计功能的系统, 以帮助设计人员在计算机上完成设计模型的构造、分析、优化和输出等工作。计算机辅助设计可提高设计的自动化程度和质量, 缩短设计周期, 借助计算机强大的计算能力, 完成一些常人难以完成的设计任务。

计算机辅助设计是伴随着计算机图形学和计算机辅助制造(CAM)技术发展起来的。20世纪50年代初, 美国麻省理工学院伺服机构实验室用Whirlwind计算机开发了第一台自动控制铣床。1958年S. Coons提出了计算机辅助设计这一概念。1962年I. E. Sutherland在麻省理工学院开发的Sketchpad人机通信的图形系统标志着计算机图形学的产生, 其方便、直观的交互方式和图形显示功能使计算机辅助设计得到了迅速发展。

计算机辅助设计的对象纷繁复杂, 涉及的范围比较广泛, 从需要满足复杂工程需求的机械产品设计(包括一般机械产品设计和汽车、造船、航空、航天等复杂产品设计)、电子产品设计、建筑设计到追求创意和美感的美术设计、广告设计、时装设计, 其中应用的专业知识、设计方法、功能需求均不相同。对于这些不同的设计领域, 计算机辅助设计系统的结构、组成、功能均存在很大差异, 但系统实现的一般性原则、原理和所采用的计算机技术却是共同的。

机械、电子、建筑是计算机辅助设计传统的应用领域, 开发技术和应用均已取得很大成功。这里主要以计算机辅助机械设计为例加以阐述。这不仅因为计算机辅助设计概念首先产生于机械产品设计领域, 而且在机械工程领域, 计算机辅助设计已形成了一些成熟的理论、技术和产品, 并得到了成功的应用。

计算机辅助设计解决的问题

以机械产品设计为例, 根据设计的各阶段工作, 计算机辅助设计要解决以下几个方面的问题:

(1) 造型 即建立设计模型。造型的主要工作是建立产品的几何形状, 输入产品的设计属性, 如物理特性、材料特性、尺寸、公差等。造型的两个技术要素是给用户设计手段和建立产品模型的表达机制。设计手段是用户用以建立设计模型的方式、方法, 如特征化、参数化技术(参见几何造型方法); 表达机制是设计模型的表示方法, 如自由曲面表达、实体表达。这两者是紧密结合的。设计手段和表达

机制的不同导致了不同类型的造型技术,如曲面造型、实体造型、特征造型等。目前先进的造型系统要求将曲面、实体、特征等多种技术融于一体,以便在计算机上建立复杂的设计模型。

(2) 分析 实现应用领域对设计对象的分析功能,如热力、静力、动力分析等。这些分析的计算量一般很大,发挥计算机强大的计算功能可以快速有效地完成分析计算。

(3) 优化 评价分析结果,优化设计模型,力图得到满足设计要求的最佳设计结果。

建模、分析、优化的过程往往需要多次循环。

(4) 输出 一种方式是绘图输出设计结果,这是目前计算机辅助设计在机械工程领域应用得最广泛、最成功的部分——计算机辅助绘图和设计。另一输出方式是把设计结果以交换文件或数据库方式传输给其他计算机辅助系统进行处理。

计算机辅助设计在系统实现和实际应用中,针对不同的应用领域和设计阶段,各自有所侧重。有的重点解决产品造型,有的在于绘图并输出设计结果,有的着重应用计算机辅助分析和优化。

计算机辅助设计采用的技术

(1) 计算机图形学 计算机图形学是计算机辅助设计中采用的重要技术。它主要包括造型、图形显示、图形标准等内容。产品几何形状的建立、表达、图示化显示等均需用计算机图形学实现。造型技术主要解决产品几何形状的表达机制和构造方法。图形显示技术是根据产品的几何形状表达在屏幕上以显示该产品的形状(参见**真实感图形生成**),图形标准主要解决所开发的计算机辅助设计系统的易移植性,它提出一组标准化的基本图形操作(参见**计算机图形标准**)。

(2) 人机交互技术 人机交互技术为计算机辅助设计提供图示化用户界面和交互数据输入机制。计算机辅助设计系统具有很强的交互性。设计模型的建立、修改等工作需要和用户进行大量交互操作来完成。系统需要不断地接收用户的输入事件,并根据这些事件迅速作出反应。人机交互技术能提供方便灵活的交互接口。交互接口的功能和性能直接影响到用户使用计算机辅助设计系统的效率和功能。

(3) 工程数据库 工程数据库为计算机辅助设计提供满足工程应用环境要求的数据管理技术。产品设计过程中涉及到大量的几何、非几何数据,这些数据的结构复杂,联系众多,需要用数据库把这些纷

繁的数据组织、管理起来,保证设计过程的顺利进行。面对工程应用领域的特殊要求,与一般数据库相比,工程数据库有其特殊的要求,如复杂对象的表达与操作、长事务管理等。

(4) 应用领域中的分析和设计方法,如有限元分析、机械设计方法等。

目前,计算机辅助设计已广泛应用于电子、建筑、机械、航空航天、汽车、造船等众多的工程领域,并取得了巨大的经济效益。计算机辅助设计的软硬件产品不断涌现,形成了一个高速发展的新兴产业。在将来的一段时期内,计算机辅助设计将主要朝着两个方向发展。一个是基于 Internet 的协同设计,基于万维网的计算机辅助协同设计系统将能够通过因特网对远程的模型进行异地设计,从而能使异地设计人员方便地交流设计思想,并能在外协件的装配等方面尽早地发现冲突,缩短设计周期。另一个是概念设计,广义上的概念设计包含了从产品的需求分析到进行详细设计之前的设计过程。它包括功能设计、原理设计、形状设计、布局设计和初步的结构设计。通过知识库和推理帮助设计人员完成概念设计,突破传统 CAD 系统只能进行辅助建模和辅助分析等的局限,赋予 CAD 系统创新设计功能,是计算机辅助设计的新的发展方向。

参考文献

1. 唐荣锡. CAD/CAM 技术. 北京: 北京航空航天大学出版社, 1994
2. 孙家广等. 计算机辅助设计技术基础. 北京: 清华大学出版社, 2000 (杨小虎 童若锋)

jisuanji fuzhu zhizao

计算机辅助制造 (computer aided manufacturing, CAM)

利用计算机对制造工厂的产品制造作业进行规划、管理和控制的过程、方法和技术。计算机辅助制造 (CAM) 与数控技术的发展密切相关,两者的紧密结合构成计算机辅助制造系统。它是计算机数控、检测技术、自动控制、成组技术、自动运输工具、机器人等各种生产技术的综合。涉及通信网络、数据结构与数据库、系统仿真、系统可靠性、数控编程、计算机辅助工艺过程编制、适应性控制等许多领域。

计算机辅助制造 (CAM) 的核心是**计算机数字控制 (CNC)**。20 世纪 60 年代末,由于计算机向小型、廉价方向发展,用一台小型计算机来控制机床很快成为现实。典型的计算机数值控制系统是靠软件

把控制逻辑的全部或一部分存储在计算机的存储器里,用以代替原来用电子电路的硬件控制。因此,CNC 系统又可称为“软联接”数控。CNC 系统的主要功能包括:①实现对机床的控制,如圆弧插补、进给量生成等;②对加工过程中发生的运动变化和误差进行动态补偿与校正;③设置诊断程序,自动寻找故障;④提供友善的菜单式面板编程和操作环境。直接数控(DNC)是用通用计算机改善传统硬连接数字控制系统性能的另一种方法。它利用一台通用计算机在分时基础上直接连接和实时控制多台机床设备,因此也称群控。DNC 系统由中央计算机、大容量存储器、通信终端、远距离数据传输系统和数字控制机床等组成。DNC 系统的主要特点是计算机可为多台分散的机床提供实时控制服务。

CAM 中数控机床加工零件的指令是由数控编程系统确定的。主要有人工编程和计算机辅助编程两种方法。人工编程仅适用于比较简单的点位控制。对于复杂的零件,采用计算机辅助编程比较合适。20 世纪 80 年代初期,随着计算机的发展,特别是图形显示设备及图形显示软件的开发,出现了交互式计算机图形数控编程系统,其主要特点是:CAM 部分可以直接从计算机辅助设计(CAD)数据库获得信息来产生刀具轨迹,并可使用户直观、清晰地看到被加工对象的形状、刀具运动过程及刀具与零件之间的干涉等,有的系统还具有加工过程仿真功能。

将计算机直接与制造过程连接起来,或者计算机本身就是加工设备的一部分,实现对制造过程的监测与控制,是计算机辅助制造的联机应用方式。在此过程中,监测是指加工过程中的采样和分析,控制则是计算机按控制指令向加工执行机构发送控制信号。采用计算机控制,还有利于实现自适应控制。除了上述计算机监控这类联机应用外,还有许多脱机应用方式。即计算机在控制过程中起支持性作用。如计算机辅助数控零件编程,计算机辅助工艺规划(CAPP),计算机辅助进行生产进度安排、提供材料需求计划等。除了计算机辅助数控编程作为 CAM 中不可分割的核心部分外,上述其他各项脱机支持,现在都已发展成可独立于 CAM 的一些计算机辅助技术了。

CAM 技术的迅速发展,已经显示出其强大的生命力和巨大的经济效益,它是增强产品竞争能力的强有力手段。采用 CAM 技术,可以减少原材料耗费,降低成本;提高产品加工精度,获得稳定的加工

质量;操作过程容易实现自动化,生产效率高,减轻了劳动强度;生产准备周期短,可大量节省专用工艺装备,适应产品快速更新换代的需要;与 CAD 衔接紧密,可直接从产品的数字模型产生加工指令,保证零件具有精确的协调和互换性;产品最后用数控测量机检验,能严格控制外形和尺寸精度。生产对象的形状越复杂,加工精度要求越高,设计更改越频繁,生产批量越小,CAM 的优势就越容易得到发挥。

CAM 系统是现代自动化制造系统的重要组成部分,在机械和其他行业中具有广阔的应用前景和强大的生命力。CAM 的发展趋势:一是由主机加终端的运行方式逐渐向分布式的工程工作站形式转变,由交钥匙系统逐渐向开放式系统转变;二是计算机辅助设计与计算机辅助制造密切结合,形成 CAD/CAM 系统;三是打破专业局限,在图形工作站上建立统一的数据库,并共享该数据库中的信息,实现一体化设计。CAD 和 CAM 的紧密结合与发展为实现计算机集成制造系统(CIMS)奠定了基础。

参考文献

1. 吴锡英. 计算机辅助机械制造. 南京: 东南大学出版社, 1990
2. Korea Y. Computer Control of Manufacturing System. McGraw Hill Book Company, 1983
(柯映林 张纪文)

jisuanji fuzhu zhiliang kongzhi

计算机辅助质量控制 (computer aided quality control, CAQ) 运用计算机实现产品、加工过程或服务质量的策划、控制、保证与改进的技术。它旨在提高质量控制的工作绩效,保证或改善产品、加工过程或服务质量,降低缺陷率或不合格率,提高顾客的满意程度,进而提升市场竞争能力。

20 世纪 50 年代,在现场质量控制中最先运用计算机。它帮助人们对现场质检数据进行统计计算、绘制和分析各种控制图,对生产过程中的质量波动状态实施有效的统计监控。由于计算机的引入,使原本由人工进行的繁杂、单调的统计工作变得简单易行,大大提高了工作效率。随着计算机技术的发展与普及,推动了质量检测设备与手段的自动化,相继出现了计算机控制的坐标测量机、测量机器人等,形成了计算机辅助检测和自动检测技术,使得质量数据的采集、传输与处理广泛应用计算机,为计算机辅助质量控制(CAQ)的形成与发展奠定了基础。此后,由于计算机辅助设计(CAD)、计算机辅助工

艺规划(CAPP)、计算机辅助制造(CAM)及计算机集成制造系统(CIMS)的发展,促进了计算机在质量管理活动中的应用范围不断扩大,不再仅仅局限于现场质量控制,而在产品设计、市场研究、供销服务等各环节也都得到了系统地应用。特别是在20世纪80年代中、后期,CAQ逐渐发展成为了覆盖产品全生命周期的综合性质量管理技术。目前,CAQ已进入了实用化阶段,相继出现了多种商品化产品,取得了良好的经济效益。

计算机辅助质量控制的基本内容包括:

(1) 质量策划 质量策划是体现预防性质量控制思想的重要步骤,也是需要在大量数据、信息或知识支持下进行综合性决策的活动,因此,利用计算机进行质量策划可提高工作质量与效率。具体包括计算机辅助产品质量计划编制,计算机辅助检测计划生成,计算机辅助前期过程质量策划等。

(2) 检测和质量数据采集 计算机辅助检测或自动检测技术具有检测效率高、测量精度高、人为误差小、能进行误差补偿和出错诊断等特点,代表当今质量检测的发展方向,成为CAQ的重要组成部分。质量数据是指导质量策划、控制、保证与改进活动的基础。主要包括反映质量水平的有关工程、生产、检验、试验等方面的数据,以质量成本表示的经济数据,顾客对产品或服务意见的数据等。借助于计算机进行数据采集,具有数据传输快、误差少、成本低、数据存储量大、便于分析处理等优点,因而得到了广泛应用。

数据采集主要有两种方式:一种是由人工检测或观察得到的质量数据、质量信息,输入到计算机存储;另一种是借助于计算机辅助检测、试验,将需要采集的非电量转化为电量,并通过放大、调整、模数转化,直接传输给计算机储存,它已成为CAQ中数据采集的主要方式。在CIMS环境下的CAQ系统常采用多台计算机组成数据采集网络或直接利用Internet来进行质量数据的采集、存储和传输。

(3) 统计分析 质量控制中最先使用计算机的目的就是用于统计分析。将统计分析中的统计计算、绘制控制图、抽样检查等内容编制成计算机软件,由计算机完成计算、分析和制图任务。这样的计算机软件可以为在通用计算机上单独运行而编制;也可以为许多专用的检测数据采集系统(如三坐标测量机、测量机器人等)运行而编制。统计分析是分析处理数据的重要工具。此外,频谱分析、时间序列分析、模式识别、专家系统和人工智能等信息处理

技术也被越来越多地用于分析质量数据,对产品、过程或服务质量状况进行评估、分析和预测,从而实施有效地控制、改进与完善。

(4) 过程质量控制 利用计算机,对贯穿于市场分析、产品设计、制造、检验、试验、销售和服务等整个业务过程的质量活动实施有效的控制。它是保证和改善产品或服务质量的最佳途径。它能及时地发现质量问题,采取措施,防止废品、次品发生,防止各种意外事件的发生,并把现场质量控制的结果反馈到质量策划层进行确认或再设计。重要的工具软件有统计过程控制软件、质量功能配置软件、试验设计软件、 workflow 管理软件等。

(5) 质量数据管理 利用数据库系统能很好地完成质量控制信息的数据采集、运用和传输,保持与其他活动的联系,使产品设计、加工、检验、控制等活动有机地联系在一起。质量数据库系统能制定质量大纲,确定“受控”参数,执行质量数据采集;可将质量手册、程序文件、操作指令、计划、工具、检验计量、鉴定要求等信息储存于计算机内,以便查询、检索;定期地对质量数据进行分析、处理,并提供图表、报告;定期地更新质量数据,有效地利用存储空间;进行有效的信息传输;提供结构控制、设计更改、修正控制、历史记录和质量预测等报告,减少重复编制质量控制程序的工作量;提供检验数据、质量损失成本以及产品缺陷及其责任者。总之,质量数据库系统具有综合和调度功能,在计算机辅助质量控制中起着十分重要的作用。

参考文献

1. Feigenbaum A V. 全面质量管理. 杨文士, 廖永平译. 北京: 机械工业出版社, 1991
2. Tannock J D T. Automating Quality Systems. London: Chapman & Hall, 1992
3. Besterfield D H. Quality Control. Englewood Cliffs, N J: Prentice Hall, 1986
4. 林志航主编. 计算机辅助质量系统. 北京: 机械工业出版社, 1997 (余忠华 吴昭同)

jisuanji guzhang zhenduan

计算机故障诊断 (fault diagnosis of computers)

通过某种测试方法与手段来检测计算机系统或电路中存在的故障,并确定其位置的过程。测试判定是否存在故障的过程称为故障检测,确定故障位置的过程称为故障定位。故障诊断是故障检测和故障定位的总称。

根据被测系统(或电路)的复杂程度和规模大小,故障诊断一般可以分为系统级、寄存器级和门级诊断3个层次。对复杂的大系统一般使用系统级的故障诊断的方法,对规模较小的电路或希望详细研究故障原因的时候可采用门级诊断的方法。无论是哪一个层次上的故障诊断其基本思想是相同的。故障检测(也称为故障测试)是故障诊断的主要任务。而故障定位则是在故障检测的基础上通过故障字典等方法来确定已被检测到的故障的位置。根据故障诊断的级别可以将故障位置确定到某一插件板(系统级诊断),某一寄存器(寄存器级诊断)或某一个门电路(门级诊断)。

故障诊断有以下3步操作:①向被测对象输入一组激励信号(称为测试向量或测试码);②接收被测对象在该组输入激励下的响应(输出)信息;③分析或比较所得到的响应输出以确定在被测对象中是否存在故障以及确定故障所在的位置等。为了实施上述第一项工作,在实施输入前必须先选取一组(或一列)适合被测对象的特定的测试码(或称测试序列),使得输入该激励信号后能从其响应输出中获取必要的故障信息。因此,这样的测试码一般来说不是随意选取的。选取测试码的过程称为**测试码生成**,这是故障诊断的核心问题。由于测试码生成的计算量很大,通常通过计算机辅助来自动生成测试码。计算机辅助自动测试系统主要包括测试码自动生成系统和测试过程控制系统。测试码自动生成系统负责用电路描述语言把被测对象的结构输入计算机,通过测试码生成算法程序自动生成测试码。测试过程控制系统负责向被测对象发送测试码,回收输出响应,显示故障情况等。

测试码生成的方法可分为确定性和非确定性两类。确定性测试码生成一般根据被测对象的结构,通过某种特定的测试码生成算法推导出检测(定位)被测对象中某特定故障的测试码以及计算出检测(定位)整个被测对象中故障的最小完全检测(定位)集。非确定性测试码生成有多种方法,随机测试码生成是其中的一种,它既可用硬件实现,也可用软件实现。

为了确定所选取的测试码的故障覆盖率(即被检测到的故障数占系统中所有可能发生故障总数的百分比)和提高测试效率,故障模拟技术也是故障诊断中十分重要的部分。从宏观上来看,故障模拟也可以把故障检测和定位的工作包括在内。就是说**故障模拟**主要可以包括:①故障检测,检验所选取

的一组激励信号是否是一组测试码,并以此测试码去检测被测对象中的全部可检出故障;②生成故障字典进行故障定位,每组测试码与其所能检出的故障作成一一对应关系,形成故障字典,在实际应用中,用于确定故障的具体位置;③对被测对象进行故障模拟分析,即将被测对象在故障条件下的操作行为和正常情况下的行为作对比和分析,并对被测对象的可测性情况作分析。故障模拟目前已是一种比较成熟的技术,并已成为提高系统可测性的有力的辅助工具,广泛应用于数字系统,特别是大规模集成电路和计算机系统的研制中。利用这种技术能有效地分析所设计电路在故障条件下的行为和可测性的情况,这对加速故障的查找和定位,缩短故障的检测维修时间,起着有效的作用。

随着被测对象的规模和复杂程度的不断增加,故障诊断的难度越来越大,测试一个一定规模的复杂电路的测试成本甚至要超过研制生产这个电路的成本,这使得传统的基于测试码生成的故障诊断技术难以付诸实际应用。摆脱这一困难的一条新途径就是采用**可测性设计**(DFT)技术,即在电路设计阶段就考虑采用一些便于测试、可降低测试成本、测试难度和测试工作量的技术和方法,从而设计出能以较小的综合代价完成预定的故障测试任务的系统。其中影响较大的主要有:

(1) 基于 R-M 展开式的组合电路可测性设计。

(2) 插入控制逻辑,通过附加硬件修改电路,可以做到只用少数几组测试码即可检测电路中所有固定故障。这类方法存在附加硬件资源(门和输入引脚)较多,电路级数多,信号时延大等缺点。

(3) 通过划分测试区,增加测试点,制订有利于电路测试的设计原则等来提高电路的可测性。这类方法统称为可测性设计专门方法。

(4) 结构可测性设计。这种设计方法可以解决包括时序电路在内的电路测试问题。其中在 1978 年由 IBM 公司提出的电平敏化扫描设计以及以后提出的边界扫描等方法是一种当今具有代表性的结构可测性设计。

参考文献

1. M Abramovici, M Breuer, and A. Friedman. Digital Systems Testing and Testable Design. Computer Science Press. 1991

2. 徐拾义. On Dependability of Digital Systems. Journal of Computer Science & Technology, 1999, 14(2)
(徐拾义 王芳雷)

jisuanji guocheng kongzhi

计算机过程控制 (computerized process control) 使用计算机系统实现生产过程的在线监视、操作指导、控制和管理的技术。这里的“生产过程”(简称“过程”)主要指工业上的连续过程和批量过程等,但也可泛指其他领域或科学实验装置中的类似过程^[1]。而“在线”指的是计算机系统与过程装置之间有直接联系,能实时地从过程输入测量信息和向它输出控制信息,使过程按操作人员的预定要求或随机要求实现变化。

20 世纪 50 年代,美国首先采用计算机自动完成对生产过程参数的巡回检测,并在此基础上试用计算机构成闭环控制系统。例如 1958 年由美国 Leeds 公司及与其合作的石油精炼厂共同研制的系统,有 160 个温度、压力和流量的测量点,30 个输出回路,平均无故障运行时间为 400 h,在石油炼制过程中起监督控制作用。此后,过程控制计算机系统的应用日趋发展,除了美国以外,在西欧、日本和前苏联先后于两三年内都在工业上成功地直接应用计算机进行自动控制。其中英国的帝国化学工业公司在 1962 年首先成功地应用直接数字控制 (DDC) 于工业生产上,直接用计算机代替模拟调节器。60 年代中期开始,过程控制计算机系统大量获得应用。

过程控制用的计算机系统在硬件规模上可以小至单片机,大至由多台中小型或超级微型计算机组成的多级分布式系统。

现代大型工业装置控制上最广泛使用的是**集散控制系统 (DCS)**,尤其具有发展前途的是采用现场总线技术的**现场总线控制系统 (FCS)**。对于需要直接置于工业过程现场工作的控制用计算机,在硬件构成及性能上应满足一些特殊要求,应选用**工业控制计算机**(简称工控机),或者按相仿的要求设计。

过程控制计算机系统的基本要求是可靠性高、控制能力强、易于使用与维护 and 性能价格比高。其中最重要的性能指标是系统的可靠性,一般采用平均故障间隔时间 (MTBF) 和系统运转率 (OR) 来描述。多台计算机通过网络相连的分散控制替代单机的集中控制,可使危险分散,系统整体可靠性提高。

过程控制计算机系统的应用软件是过程控制计算机系统设计人员针对特定生产过程编制的监视、控制和管理用软件,按其功能可划分为数据采集、过

程控制及管理、过程输出、人机接口、打印输出等功能模块。其中过程控制及管理模块是应用软件的核心,是各种控制及管理策略的具体实现。

为使现场的控制和管理人员能针对特定的生产过程方便而高效地设定和更改控制、管理的策略和输入输出方法,应用软件通常不是由计算机程序员编制,而是由熟悉生产过程的人员通过“组态”方式生成。即只要通过在计算机画面上将一些表示策略或算法的模块相连接,或者通过在表格上填写一些参数或回答“是”或“否”后,再经过计算机编译即可生成或修改所需的应用软件。为实现这类功能需要有相应的组态软件包,它可由计算机系统供应商提供,也可是某些在一定范围内通用的商品化软件。

计算机参与过程控制的方式可以大致分成 4 种类型:操作指导、直接数字控制、监督控制和分散控制。在已具备必要的自动化仪表(包括传感器、变送器、在线分析仪和执行器等)和过程控制计算机系统的前提下,实现计算机过程控制还应包括下列步骤:①建立控制对象的数学模型;②确定控制方案和选择合适的控制算法;③编制控制软件;④系统的单元调试和联调;⑤安全、平稳的投运策略。

过程控制计算机系统在应用于连续过程或批量过程上有不同的侧重点。

较早应用计算机控制的是连续过程,例如在石油炼制、化肥生产、发电等行业中的精馏、蒸发、蒸汽锅炉给水、炉内燃烧等过程。这些过程的启动、停车操作与它们的稳定运行阶段在时间上相比显得很短,因此主要考虑在它们的稳定运行阶段实现计算机的闭环自动控制。对过程的模型化,过程控制回路的参数整定,各种复杂或高级控制算法的运用等,也主要针对其稳定运行阶段。而对启动和停车操作,则采用手动操作方式实现。因此对各个控制回路应能进行“自动”方式和“手动”方式的无扰动切换。

对于批量过程,例如零件加工、产品包装、金属部件热处理、生物发酵、罐头食品杀菌、间歇釜聚合反应等,它们的工艺操作过程是完整的从开始到结束的相类似循环。每一循环由若干不同工作状态的阶段组成。这里还要再区分两类情况:一类是如零件加工、产品包装等简单的顺序加工过程,在它们的操作循环中从上一阶段到下一阶段的转变,只是简单地按时间顺序或满足某些机电连锁条件即可,也就是**顺序控制**问题。而另一类如罐头食品

杀菌、生物发酵、间歇釜聚合反应等比较复杂的热力过程或甚至伴有化学反应、生化反应的过程,它们在不同工艺阶段往往具有不同的动态特性,也是一类具有非线性和时变特征的对象。这类过程从上一阶段到下一阶段的转变,不仅要考虑时间和连锁条件等因素,还必须满足某些过程参数的要求。目前还只能对少数这类过程实现较满意的计算机控制,而且常常要使用专家系统、模糊控制和其他的人工智能策略。

[注] 另外,有一类主要出现在制造行业的工业过程,如飞机、汽车、机床、家用电器等的流水线方式加工过程,这种控制称为“生产线控制”,一般采用“顺序控制”方式。它们的控制输入和输出,主要是数字量或开关量。目前在现代化的制造业中,多采用以可编程控制器为现场控制级的集散控制系统,实现计算机控制。

参考文献

王锦标, 方崇智. 过程计算机控制. 北京: 清华大学出版社, 1992 (赵鹏程)

jisuanji guocheng kongzhi fangshi

计算机过程控制方式 (computerized process control manner) 使用计算机参与过程控制的方式。它大致分为 4 种类型: 操作指导、直接数字控制、监督控制和分散控制。

操作指导 应用计算机进行过程的实时数据采集、处理并输出操作指导信息, 指导人工操作的控制方式称为操作指导。

操作指导方式 的优点是简单、比较灵活和安全。计算机的输出部分与生产过程的各个控制单元并不直接联系, 而是由操作人员按照计算机给出的操作指导信息加以判断后, 人工去改变各个控制器设定值或直接操作执行器。因此, 它实际上是一种开环结构。但由于人工参与分析判断和操作, 受人的生理条件约束, 控制的实时性和精确度受到限制。

操作指导方式常用于对控制规律尚未确定的场合, 及计算机控制系统设置的初期阶段或新的控制软件的调试阶段等。

直接数字控制 实时采集数据, 按照一定的控制策略计算产生控制指令, 并通过输出通道直接控制生产过程的控制方式称为直接数字控制 (DDC)。系统的构成如图 1 所示。直接数字控制属于计算机闭环控制, 是计算机控制最普遍的一种应用方式。为充分发挥计算机的利用率, 一台计算机通常采用分

时方式控制多个回路。

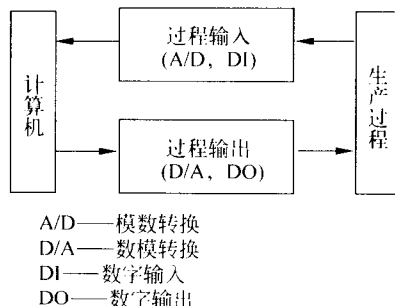


图 1 直接数字控制系统框图

直接数字控制不仅能完全取代模拟调节器, 实现多回路的比例积分微分 (PID) 调节, 而且无需改变硬件, 只要改变程序就能有效地实现各种复杂的控制策略, 如前馈控制、非线性控制、自适应控制等。

直接数字控制有以下两种设计方法:

(1) 模拟化设计 即对直接数字控制系统按连续系统理论设计校正环节 (调节器), 然后数字化 (离散化) 的方法;

(2) 直接数字设计 即按离散系统理论进行数字调节器的直接设计。

实现直接数字控制的计算机直接承担控制任务, 这就要求其可靠性高、实时性好和适应性强。由于工业生产现场环境恶劣、干扰频繁, 必须采取各种抗干扰措施或冗余设计等, 以提高系统的可靠性。

监督控制 通过计算机系统的过程输入通道实时采集过程信息, 按照软件中包含的过程数学模型和一定的控制算法运算后, 自动地改变控制回路设定值 (即模拟调节器或计算机数字控制回路的设定值), 从而使生产过程始终处于某种最优工况的计算机控制方式称为监督控制。因其工作方式是改变设定值, 也称为设定值控制 (SPC) 方式。

监督控制的效果, 主要取决于过程数学模型与控制算法的优劣。控制系统中需考虑的因素越多, 对象数学模型越复杂, 对所用计算机的计算能力、存储容量等性能的要求也越高。在实现监督控制的过程中, 因涉及系统建模与优化, 应用软件开发的工作量较大。

分散控制 与空间位置分散但其输入输出之间互有关联的多个被控对象相对应, 将多台计算机分散分布, 就地实现数据采集和自动控制, 同时经过通信网络将这些计算机设备连成一体, 实现控制上

分散、操作管理上集中的一种控制方式称为分散控制。

分散控制方式出现于 20 世纪 70 年代中期,它是在计算机、通信、控制和显示技术的基础上产生的。采用分散控制方式的过程控制计算机系统也称为**集散控制系统**。它由多种专用计算机、外部设备和通信网络设备等构成,通常构成一个多层(或称多级)的网络系统。

与早期采用单一的中型或大型计算机的集中控制方式相比较,分散控制方式具有以下一些优点:

(1) 可靠性高 由于系统功能分散在各台计算机上实现,因此某一计算机出现的故障不会导致系统其他功能的丧失。此外,由于系统中各台计算机所承担的任务比较单一,因此可针对需实现的功能采用具有特定结构和软件的专用计算机,从而使系统中每台计算机的可靠性也得到提高。

(2) 扩充性好 系统中各台计算机通常采用联网方式进行信息传输,当需要改变或扩充系统功能时,可将专用计算机方便地连入系统通信网络或从网络中卸下,而不影响或较少影响系统中其他计算机的工作。

(3) 易于维护 功能单一的小型或微型专用计算机的维护比大型通用计算机的维护简单、方便,对维护人员的专业化要求也较低。

参考文献

1. 谢剑英. 微型计算机控制技术. 北京: 国防工业出版社, 1985
2. 王锦标, 方崇智. 过程计算机控制. 北京: 清华大学出版社, 1992
3. 顾兴源. 计算机控制系统. 北京: 冶金工业出版社, 1981 (赵鹏程)

jisuanji jifang sheshi

计算机机房设施 (computer room facility)

保证计算机在机房安全可靠运行所配置的各种设施。计算机机房指安装各类计算机系统及相关设备的专用场所。

机房环境条件

温度与湿度 温度与湿度是计算机机房最重要的环境条件,对计算机可靠运行和寿命均有显著影响。温度偏高导致元器件散热困难,容易造成元器件损坏及工作不稳定,而且加速元器件的腐蚀和磨损而影响其寿命。温度过低则导致部分敏感的机械部件由于膨胀系数不同而运转失误。温度梯度也需

控制在一定范围,因温度变化速度过快会造成某些敏感部件的运转失误。温度条件还应考虑机房工作人员舒适性需要。为此,大多数机房规定的温度范围为 $18 \sim 25^{\circ}\text{C}$, 冬天可靠近下限,夏天则靠近上限。温度梯度的范围大致为 $5 \sim 10^{\circ}\text{C}/\text{h}$ 。

湿度过高容易造成高密度组装的元器件表面或接插件的导体之间短路,还会使某些易吸湿物质(如纸张)运动不顺畅。湿度过低会使机房中容易产生静电高压,不仅影响计算机正常运行,还易损坏集成电路。机房适宜的相对湿度为 $50\% \pm 10\%$ 。

洁净度 洁净度指空气中单位体积内的尘埃含量,它也是计算机机房的重要指标。灰尘对磁盘机、磁带机及软磁盘机等精密机械会造成严重危害。例如硬磁盘机磁头和盘片之间的缝隙只有零点几微米,如缝隙中沾染了灰尘就可能发生存取数据错误并会导致磁头磨损或划伤盘片。灰尘如积聚在高密度电路表面或接插部位,则既容易造成吸湿短路也可能引起接触不良。空气过滤器中积聚过多灰尘则导致通风不畅使机器过热,机械转动部分落入灰尘会使设备磨损加剧。机房洁净度以单位容积中尘埃粒数衡量, A 级机房要求此数小于或等于 $3\,500 \text{ 粒}/\text{ft}^3$, B 级机房要求此数小于或等于 $10\,000 \text{ 粒}/\text{ft}^3$ (尘埃粒径大于或等于 $0.5 \mu\text{m}$, $1 \text{ ft}^3 = 0.0283 \text{ m}^3$)。

静电与电磁干扰 静电是由于磨擦生电、感应或辐射造成的电荷积累产生的一种高压电场。而计算机设备所包含的半导体器件如 MOS 电路及其他高输入阻抗电路对静电非常敏感,这些电路易受静电干扰导致工作失效,甚至被静电高压损坏。因此机房应采取防静电措施。

各种来自外界(如雷电、邻近的高强度无线电波及强高压电器设备)的电磁干扰会影响计算机中一些高灵敏度的弱信号放大器正常工作。机房应采取措施予以防范,如确保设备接地良好,必要时对某些关键装置进行屏蔽防护。按照国标 GB 2887—82 的规定,机房内无线电干扰场强,在频率范围为 $0.15 \sim 500 \text{ MHz}$ 时应不大于 126 dB , 磁场干扰场强不大于 $800 \text{ A}/\text{m}$ 。

电源与接地 机房必须有良好的供电系统以保证计算机正常运行。对于必须连续不间断运行的计算机系统应采用双路供电的方式,避免任何一路因偶然故障导致系统瘫痪。机房的供电交流电源应有较高的电压和频率稳定度。按国标 GB 2887—82 规定, A 级电压相对误差范围是 $-5\% \sim +5\%$, B 级电

压相对误差范围是 $-10\% \sim +7\%$, C 级电压相对误差范围是 $-10\% \sim +10\%$; A 级频率误差范围是 $-0.05 \sim +0.05 \text{ Hz}$, B 级频率误差范围是 $-0.5 \sim +0.5 \text{ Hz}$, C 级频率误差范围是 $-1 \sim +1 \text{ Hz}$ 。

计算机机房有多种接地线,即交流电源地线、直流电源地线及安全接地线等。这些地线有的可以共用,有的则必须分开。共用地线简便但可能造成交流信号对直流地线的干扰而影响计算机正常工作。目前较常用的方法是直流地与安全地共用而与交流地线分开以避免干扰。接地电阻数值应尽量小,一般直流地与安全地的接地电阻小于 $1 \sim 2 \Omega$,交流地接地电阻小于 $2 \sim 4 \Omega$ 。

计算机机房设施

空调通风设施 为保持机房的洁净度并将温度及湿度控制在特定范围内,机房采用封闭的环境并借助空调设施进行通风。目前大多数计算机机房都安装了专用空调装置,以便根据季节变化及计算机的散热量灵活调节空调机容量,使机房温湿度控制在所需范围内。

不间断电源 不间断电源(UPS)具有稳频稳压功能,供电线路通过它将交流电送到计算机,不仅提高了电源稳定性而且当外界供电突然中断时,可借助后援电池保持系统继续供电一段时间(根据系统的需要设置,通常为十几分钟到几个小时),使维护人员可及时采取措施保存系统运行的现场状态信息和其他重要数据,保证系统信息的安全性及完整性。

防静电地板 机房地面宜采用在建筑物原有地板上架高 300 cm 左右的防静电地板,这样可防止在机房中产生严重的静电高压,而且地板下的空间可用作通风回路和各种电源、信号线电缆以及地线的铺设路径。

防火设施 计算机机房四周墙壁及门窗、天花板均应采用防火材料。必要时装置高温和烟雾敏感探测装置和自动报警设备,对机房各个角落进行探测,以便一旦出现可能引起火灾的因素能及时发觉并采取措施予以消除。条件许可时应装置自动灭火设施。

照明 照明是提供工作人员维护计算机的基本条件,照度应按照 GB 2887—82 标准执行。除正常照明设施外,机房应设置事故照明装置,一旦供电中断时保证机房照明。

其他设施 其他必要的辅助设施包括:保证机房洁净度及清新空气的装置(如进入机房新空气的

过滤装置、负离子发生器等),监测机房环境条件的装置(如连续型温湿度记录仪),防灾装置(如防地震的机柜锁定装置、防电击的避雷设施、防止过湿和漏水的水敏传感器以及紧急灭火设施等)。

今后的计算机机房设施将朝智能化方向发展,各种环境条件的提供与保证均采用自动监测、自动控制。考虑到计算机系统本身的脆弱性及计算机犯罪的严重性,计算机机房的相应的安全保险设施将日趋完善。

参考文献

劳诚信,姜国雄等. 计算机安全指南. 北京:清华大学出版社,1993 (苏振泽)

jisuanji jicheng zhizao xitong

计算机集成制造系统 (computer integrated manufacturing system, CIMS)

制造型企业为适应市场激烈竞争而采用的一种生产组织与管理模式。它综合运用计算机信息处理技术和生产技术,对制造型企业经营的全过程(包括市场分析、产品设计、计划管理、加工制造、销售服务等)的活动、信息、资源、组织与管理进行总体优化组合,以提高企业的竞争能力,使企业获得最大的总体效益。

发展简史

计算机集成制造(CIM)的概念是美国 J. Harrington 于 1973 年提出的。这一概念的形成一方面是由于企业外部环境的变化——竞争日益加剧,另一方面是计算机技术、信息处理技术、自动化技术的发展为 CIM 的实现提供了技术可能性。

计算机集成制造(CIM)的技术来源于 3 个方面:计算机辅助设计、计算机辅助管理及制造自动化。在计算机辅助设计(CAD)方面,美国于 1963 年研制成功第一个几何造型 CAD 系统,20 世纪 60 年代末,计算机辅助工艺规划(CAPP)开始研究。随后,计算机辅助制造(CAM)、计算机辅助工程(CAE)也陆续得到发展。在计算机辅助管理方面,1968 年美国 IBM 公司提出了第一个生产信息与管理系统(PICS)。其后,出现了功能日益完善的制造信息系统(MIS)。在制造自动化方面,其发展历程可追溯到第一台数控机床(参见计算机辅助制造)的诞生。但是可供企业使用的数控机床产品直到 60 年代中期才在市场上出现。在此基础上,1967 年,第一套柔性制造系统(FMS)研制成功,70 年代逐渐形成工业应用产品。70 年代以来,制造业面临着日益激烈的市场竞争的外部环境。为了缩短产品

的生产周期,提高产品质量,降低产品成本,上述三方面的集成日益受到重视,陆续实现了 CAD 与 CAM、CAD 与 MIS、CAPP 与 MIS 等局部系统的集成。80 年代以来,企业对信息集成提出了更高的要求。随着计算机技术、计算机网络技术、数据库技术等信息技术迅速发展,CIM 进入了迅速发展时期。它在概念和内涵上都得到了很大的丰富和发展,如并行工程(CE)、即时制造、精良生产、敏捷制造、经营过程重组等新思想、新概念不断产生。企业从 CIM 的应用中也获得了巨大效益。

计算机集成制造系统(CISM)的构成

CIMS 一般可分为 4 个功能分系统(即经营管理信息分系统、产品设计自动化分系统、制造自动化分系统和质量保证分系统)和 2 个支撑分系统(即计算机网络支撑分系统与数据库支撑分系统)。4 个功能分系统通过 2 个支撑分系统有机地实现集成。下面对该 6 个分系统简要地加以说明。

(1) 经营管理信息分系统 它以制造资源规划(MRP II)为核心,具有包括市场预测、经营决策、各级生产计划、生产技术准备、销售、供应、财务、成本、设备、工具以及人力等信息管理功能。

(2) 产品设计自动化分系统 它是用计算机来辅助产品设计、工艺规划及制造,即常说的 CAD/CAPP/CAM 系统以及产品数据管理系统 PDM。

(3) 制造自动化分系统 它是 CIMS 中信息流和物料流的汇合点。不同的制造企业具有不同的制造设备和制造环境,其自动化水平也有很大的差别。如离散型制造,它可能由数控机床、加工中心、清洗机、测量机、各类运输工具、各类仓库等组成。这些与一般制造系统没有很大的不同。它们的区别在于,在 CIMS 中,这些设备通过计算机网络及相应支持软件集成在一起,可根据产品的工程技术信息及生产计划,灵活而高效地完成作业调度和制造任务。

(4) 质量保证分系统 包括质量决策、质量检测与数据采集、质量评估、控制与跟踪等功能。

(5) 数据库支撑分系统 它是覆盖企业全部信息以支持 CIMS 各功能分系统的数据库管理系统。逻辑上的统一性,物理上的分布性是其基本的特征。

(6) 计算机网络支撑分系统 它是支持 CIMS 其他 5 个分系统的网络通信系统。一般采用国际标准或工业标准规定的网络协议,支持资源共享。开放、异构环境互联是企业计算机网络支撑分系统基本要求。

我国 CIMS 的进展

CIMS 在我国从 1986 年起作为高技术发展计划(亦称为 863 计划)中的一个主题开始实施。1987 年第一个 CIMS 技术研究实验基地——CIMS 实验工程开始建设,经过 5 年努力,于 1992 年完成。1994 年获得美国制造工程师协会(SME)颁发的大奖领先奖,表明我国 CIMS 技术达到国际领先水平。1989 年开始,陆续有不同行业的多家企业实施 CIMS,大多获得了明显的效益。1995 年,北京第一机床厂获得了美国制造工程师协会颁发的工业领先奖。

参考文献

863/CIMS 信息网. 计算机集成制造系统 CIMS 问答. 北京: 兵器工业出版社, 1993 (肖田元)

jisuanji keweihuxing

计算机可维护性 (computer maintainability) 系统失效后在规定的时间内可修复到规定功能的能力。衡量计算机可维护性的指标称为可维护度,记为 $M(t)$ 。一个计算机系统的可维护度 $M(t)$ 是指该系统失效后,在规定的時間间隔 t 内被修复的概率。在系统的失效率和修复率都是常数的情况下,根据可维护度的定义,可得可维护度 $M(t)$ 与修复率 μ 的关系为

$$M(t) = 1 - e^{-\mu t}$$

修复率(repair rate)表示在单位时间内完成修复的概率,亦即反映系统可维护性高低的一个常数,系统可维护性的另一个重要参数是平均修复时间(MTTR)。它是指对失效的系统进行修复所需的平均时间。平均修复时间与修复率的关系为

$$MTTR = \frac{1}{\mu}$$

可见,修复率 μ 对于一个系统的可维护性至关重要。修复率除了依赖于维修人员的技术水平外,还依赖于维修的环境与条件。

平均修复时间 MTTR 可用实验方法求得: 对 1 个系统注入不同的故障,每注入 1 个故障就进行修复并记下修复时间,最后可求出平均修复时间 MTTR。为了使所求得的 MTTR 比较准确,在进行实验时应注入足够多的不同类型的故障,并让不同技术水平的维修人员进行修理。

在实际工作中,一般将维修分成 3 级。

第一级维修为现场级,包括所有能在系统所在地(如计算机房)进行的维修。通常现场级维修要

求将故障定位至电路板级,然后更换有故障的电路板,从而使系统恢复正常运行。由于不可能将复杂的故障检测与定位仪器带到现场,现场级维修通常要借助于系统所提供的内装测试(built-in testing)功能。

第二级维修为中间级。在现场不能实施的维修可在邻近现场的维修点进行,称为中间级维修。通常在维修点具有比在现场更好的维修设备及更充足的备件,因此中间级维修能解决在现场不能解决的问题。另一方面,由于维修点比工厂更靠近现场,因此中间级维修比起将失效系统送回工厂维修要简便省时。

第三级维修为工厂级。现场级维修及中间级维修不能解决的问题只能由生产工厂来维修。这时失效的系统或部件被送回工厂进行维修。

上述3级维修的修复率或平均修复时间通常相差很大。用于计算可维性的MTTR是现场级维修的平均修复时间,它远小于中间级维修的平均修复时间,而中间级的平均修复时间又远小于工厂级维修的平均修复时间。

为了寻求提高系统可维护性的途径,有必要进一步分析对失效系统进行修复所需时间的组成。一般可将系统修复时间分为两大部分:被动修复时间和主动修复时间。被动修复时间指从发现系统存在故障至维修人员开始进行维修的这段时间。其长短取决于管理水平和技术支持能力。主动修复时间是指维修人员实际进行修理所花费的时间。它又可进一步分为下列几部分:①故障检测时间,即用测试手段确认存在故障所需的时间;②故障定位时间,即用测试手段确定故障位置所需的时间;③修理时间,即更换故障部件或修复故障部件所需的时间;④验证时间,即验证系统或部件确实已被修复、能正常工作所需的时间。主动修复时间与维修人员的技术水平有重要关系,但另一方面可通过改进系统的硬件与软件设计,完善故障检测和诊断措施来达到缩短主动修复时间的目的。

为了提高系统的可维护性,一方面应提高管理水平和改善后勤支持,以缩短被动修复时间;另一方面,应在设计系统时就考虑使系统易于测试、诊断和修理,包括采用自动诊断技术、可测试性设计技术、内装自测试技术以及系统结构的模块化技术等,以缩短主动修复时间。

参考文献

1. Johnson B W. Design and Analysis of Fault-

Tolerant Digital Systems. Reading, Massachusetts: Addison-Wesley Publishing Company, 1989

2. Kraft G D, Toy W N. Microprogrammed Control and Reliable Design of Small Computers. Englewood Cliffs, New Jersey: Prentice Hall Inc., 1981

(胡谋)

jisuanji keyongxing

计算机可用性 (computer availability) 计算机系统在某一时刻能提供有效使用的程度。其度量指标称为可用度,用A表示。可用度是在任何指定时刻系统能正确运行的概率。

一般情况下,系统发生故障后是可以修复的,并且其失效率和修复率都是常数。这时可用度A用下面的公式计算:

$$A = \frac{MTTF}{MTTR + MTTF}$$

式中 MTTF——平均无故障时间;

MTTR——平均修复时间。

平均无故障时间是指在相当长的时间内,从系统正常启动或一次故障修复后重新运行到下一次失效的平均时间。

平均修复时间是指在许多次故障中,从开始失效到修复故障所用的平均时间。故障修复时间可用下面的公式来计算:

$$\text{故障修复时间} = \text{申请维修时间} + \text{等待时间} + \text{探查时间} + \text{恢复时间}$$

式中 申请维修时间——开始失效到报告维修人员的时间;

等待时间——请求现场服务,等待维修人员到达和备品配件到位的时间;

探查时间——进行故障检测、诊断和修复所需的时间;

恢复时间——系统重新启动并开始算题所需的时间。

可用性与可靠性不同,在某时刻以前系统可能出现过任何数量的故障,但只要所有故障均已修复,该系统仍是可用的。但对可靠性来说,只有在某时刻以前没有出现任何故障,该系统才被认为是可靠的。

在早期的计算机系统上人们就已经注意到统计可用性数据的问题。随着并行处理技术的发展,计算机体系结构日趋复杂。在这些系统中,当发生局部失效时,系统可按降级模式继续运行,但吞吐量降

低了。为了统计系统的停机时间,引入了加权系数的方案。在该方案中,对硬件和软件的每一项均给以加权系数 $\alpha(0 \leq \alpha \leq 1)$ 。若为系统失效,则 $\alpha = 1$;若为局部失效,则 $0 \leq \alpha < 1$;若故障部件有热备用部件进行自动更换,则 $\alpha = 0$ 。计算停机时间,还应乘以加权系数。所谓系统失效是指由于系统主要部件(如电源,CPU)发生故障或由于软件原因导致整个系统不能工作。所谓局部失效是指系统中只是某些部件发生故障,而系统的其余部分仍能正常运行的情况。此外,加权系数数值的确定依赖于特定的应用,它应能反映该项故障引起的吞吐量的损失。

提高计算机可用性的途径,一是提高计算机的可靠性,二是提高计算机的可维性,完善故障诊断与测试技术以及系统恢复和部件更换技术,提高维护人员的素质也是不可忽视的因素。

故障测试与诊断技术用于查找系统或电路中的故障,确定有无故障及故障的位置。故障测试有多种方法与手段,例如:故障模拟、自动测试码产生等。通过较快地诊断故障而增加系统的可用性。

系统恢复技术允许进程在不丢失或丢失少量信息的情况下,退回到错误发生前的某一点,重新开始执行而不必重启系统,恢复技术通常用软件来实现。检测故障后还可简单地切除故障器件,但此时系统的能力降低,这种处理叫做缓慢降级。例如,在计算机中存储器的某些部分当其有故障时,可以把它们从地址空间中切除,这通常借助于硬件和操作系统中的虚存管理机制来实现。对于多处理机系统,切除某些故障的处理机后,系统必须重新组合,并把任务重新分配给有效的处理机,使系统继续运行,并可同时对故障的处理机进行维修。

部件更换技术直接影响系统的可用性。为此,计算机应采用结构模块化与功能模块化的部件来构造。结构模块化即元器件应装配成可替换的模块形式。例如,计算机的存储器装配在可更换的插件板上。功能模块化即制造程度更高的较大模块。例如,计算机的CPU模块,以利于减少模块种类。这样维修人员更换模块十分方便,从而减少了故障修复时间。

故障修复中的等待时间与所提供的维护服务有密切的关系。计算机厂家或销售部门应为用户提供良好的维修服务,并加强对用户的培训。计算机用户也必须提高技术、加强管理,使计算机系统的可用性不断提高。

参考文献

西沃赖克 D P, 斯沃兹 R S 著. 可靠系统的设计理论与实践. 袁由光, 曹译翰, 刘志模, 陈以农译. 北京: 科学出版社, 1988 (吴正凝)

jisuanji kongzhi

计算机控制 (computer control) 以数字计算机为核心控制部件并借助一些辅助装置与被控对象相联系, 以达到特定控制目的的过程和技术。这里的数字计算机可以有各种规模, 包括从微型到大型的通用或专用计算机。辅助装置主要指计算机输入输出接口、检测装置和执行装置等, 它们可以是模拟设备、数字设备或是数模混合设备。与被控对象的联系和部件间的联系, 可以有有线方式, 如通过电缆的模拟信号或数字信号进行联系, 也可以是无无线方式, 如用红外线、微波、无线电波、光波等进行联系。被控对象的范围很广, 包括各种生产过程、机械装置、交通工具、机器人、仪器仪表、家用电器和儿童玩具等。控制目的可以是使被控对象的状态或运动过程达到某种要求, 也可以是为了达到某种最优化目标。

与一般的自动控制相同, 计算机控制可以是闭环的, 典型的就是直接数字控制方式, 这时计算机要不断采集被控对象的状态信息, 按照一定的控制策略和步骤处理后, 输出控制信息直接影响被控对象。它也可以是开环的, 这有两种方式: 一种是计算机只按时间顺序或某种给定的规则影响被控对象; 另一种是计算机将来自被控对象的信息处理后, 只向操作人员提供操作指导信息, 然后由人工去影响被控对象。

在上述的几种控制方式中, 闭环方式是用得最普遍、最基本的控制方式。上述第一种开环方式主要用在顺序控制系统, 如机械加工、设备包装等行业。后一种开环方式也称为操作指导方式, 就其信息传送回路来说, 由于人的介入, 实际上有一定程度的闭环反馈作用。由于加入了人的智能因素把关, 因此比较灵活和安全。在对安全性要求较高、被控对象不确定因素较多及快速性要求不高的场合, 例如某些复杂生产过程的控制中用得较多。

将数字计算机用作控制器的思想萌生于 20 世纪 50 年代初。最早, 试图在飞行器控制中应用计算机, 但由于当时的通用机体积大、能耗多、可靠性差, 因此使用了专用机——数字微分分析器。

此后, 计算机控制的主要进展是在工业过程控

制领域(参见计算机过程控制)。美国首先用计算机实现过程的巡回检测和数据采集。20世纪50年代后期又成功地实现了计算机在线闭环控制。1962年英国率先实现了计算机的直接数字控制,充分利用计算机的高速分时运算能力代替多台控制功能较简单的常规模拟控制仪表。

20世纪70年代微型计算机的诞生,使计算机控制的应用进入一个新阶段。在工业过程控制领域,不久就出现了控制上分散、操作管理上集中的分散控制系统,也称集散控制系统(DCS)。如美国Honeywell公司于1975年推出的TDC-2000系统。集散控制系统在结构上是由多台计算机及其他设备用通信网络联系在一起的分级分布式计算机控制系统,它有效地解决了直接数字控制方式计算机多回路集中控制带来的危险集中问题。直至90年代,集散控制系统一直是世界各国也包括我国在过程计算机控制方面的重要发展方向,世界上已有数万套以上的集散控制系统在现场运行,取得了可观的经济效益。这一年代出现的现场总线控制系统(FCS)是更具有发展前景的一种分级分布式计算机控制系统。微型计算机控制技术还迅速渗透到机电控制领域,不仅在航空航天、军事装备、机器人、机械生产自动线中得到广泛应用,而且在家庭生活设施中,如洗衣机、微波炉、空调机等,都有由微型计算机(这里是单片计算机)构成的控制系统。

20世纪80年代后期出现的嵌入式技术及其产品,无论是嵌入式微处理器、嵌入式组件(如PC-104总线系统),或是嵌入式操作系统(如Nucleus)等,都已在计算机控制中获得了广泛应用。

计算机控制与模拟控制器控制相比较,就执行功能来说都包括被控对象状态信息检测、信息加工决策和控制信息输出这三部分,但模拟控制器的控制过程在时间上是连续并行进行的,而计算机对信息的加工是离散串行的,且计算机内也只能处理离散的数字信息。

在控制理论中,计算机控制系统归属于离散控制系统,它是指时间上离散或采用断续控制方式的一类控制系统,也称离散时间控制系统或采样控制系统。它的特点是系统内的信号仅在特定的离散瞬时表现为时间的函数。在自动控制技术的发展初期,一些控制系统由于采用了断续工作的机械式测量或控制元件,导致离散控制系统理论的产生和研究。数字计算机被引入控制系统后,由于其处理的信息在时间上是离散的,因此在系统设计时,需经离

散化处理才能成为离散控制系统。

离散控制系统中的被控参数测量、控制算法的计算及控制信号输出均是在离散的时间点上进行的。测量信号的离散化过程称为采样过程,即以给定的周期(称为采样周期)在离散的瞬间把连续信号转换成脉冲序列。为了理论计算和数据处理的方便,需要在每一次采样之后使采样输出信号保持不变。在计算机控制中,测量信号的采样过程由采样器和模数转换(A/D)器件实现。图1给出了信号的采样过程,其中 $f(t)$ 为原始的测量信号, $f_s(t)$ 为对 $f(t)$ 采样的脉冲序列, $f_h(t)$ 为对 $f_s(t)$ 作保持处理后最终的阶梯型采样输出信号, T_s 为采样周期。采样周期的选择对于系统性能有极大的影响。选择采样周期的理论依据是香农(Shannon)采样定律。该定律指出:如果一个时间信号 $f(t)$ 中不包含频率大于 f_{\max} 的谐波分量,则采样频率(即采样周期 T_s 的倒数)必须大于 f_{\max} 的两倍,才能从采样输出信号 $f_h(t)$ 中复原 $f(t)$ 。但在实际应用中,由于测量信号的最大谐波频率难于确定,因此采样周期的选择问题大多通过试验来解决。

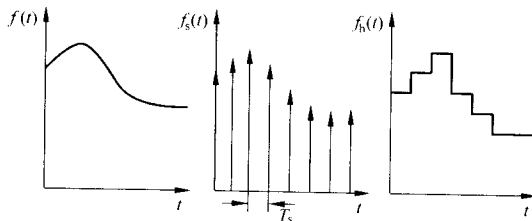


图1 采样过程

为了适应离散控制系统中被控参数的这种采样测量机制,在控制策略的设计和计算中需采用离散时间数学模型。基本的离散时间数学模型是差分方程。它表示了相邻的离散瞬间系统中各变量之间的关系。对于线性的离散控制系统差分方程模型,还可以通过Z变换法将其转换为代数方程,使离散控制系统的分析和设计工作得到简化。

由基于被控对象的离散时间数学模型设计出来的控制规律,通常也表示为在离散瞬间的时间函数。与测量信号的采样过程类似,这些在离散时间点上的控制作用也需通过保持处理变换为阶梯型的控制信号,然后作用于被控对象。在计算机控制系统中,这一过程是由数模转换(D/A)器件完成的。

与连续控制系统相比,离散控制系统具有数字系统的一些独特优点,如:①在很多场合,其结构比

连续系统简单;②信号的传递和转换有较高的精度;③抗干扰性能较好;④适于采用数字计算机作为控制器,可以一定的精度高速完成复杂的计算任务,易实现对于复杂被控对象的高级控制策略(例如预测控制、自适应控制、优化控制等)。

在计算机控制实施时,通常由控制部分和被控对象构成计算机控制系统。其控制部分除了图2所

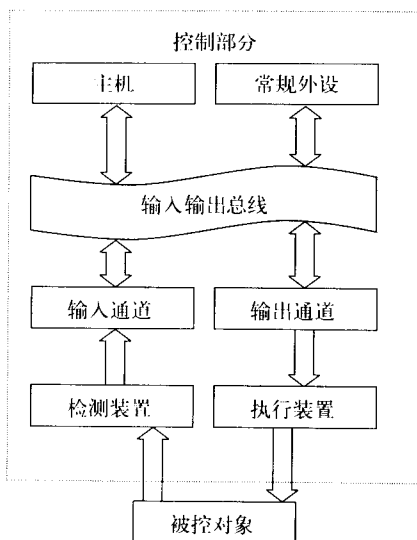


图2 计算机控制系统组成框图

示的硬件部分,还包括软件部分。计算机控制软件包括系统软件和应用软件。系统软件一般包括操作系统、语言处理程序和服务性程序等,它们通常由计算机制造厂为用户配套,有一定的通用性。应用软件是为实现特定控制目的而编制的专用程序,如数据采集程序、控制决策程序、输出处理程序和报警处理程序等。它们涉及被控对象自身的特征和控制策略,也涉及控制部分中相应部件的特性等,由实施控制系统的专业人员自行编制。对于不同规模的计算机控制系统,例如打包机的单片机控制系统和大型化肥生产装置的集散控制系统,硬件和软件部分可以差异很大。

由于使用计算机作为控制工具,控制系统及控制在深度和广度上都在不断发展。在广度上,向着大系统或系统工程方向发展,从单一过程、单一对象的局部控制,发展到对整个工厂、企业,甚至对社会经济、国土利用、生态平衡等大规模复杂对象进行控制,实现控制和管理一体化。在深度方面,则向着智能化发展,不仅引进自适应、模糊决策、神经网络、自学习等控制方法,而且模拟人的视觉、听觉和触觉,能识别文字、图像、言语、物体,根据感知的信息进行直观判断、推理分析、自行学习解决问题,获得更高层次的控制效果。

参考文献

1. 王锦标,方崇智. 过程计算机控制. 北京:清华大学出版社,1992
2. 冯培梯. 计算机控制技术. 杭州:浙江大学出版社,1990 (褚健 赵鹏程 吴铁军)

jisuanji leixing

计算机类型 (computer category) 按某个或某些特征将计算机分成的若干种不同的类型。不同的分类方法使用不同的分类特征,得出不同的计算机类型。

计算机分类的方法和所分成的类型是随着计算机技术和应用的进展而不断变化的。早在20世纪50年代,已开始对电子计算机进行分类,当时把计算机按应用对象划分为科学计算用计算机和商业数据处理用计算机两类。前者采用二进制,有浮点运算,字长固定;后者用定点十进制运算,字长可变。60年代中期,由于出现了通用系列计算机,使上述两类计算机的界限消失,计算机开始按性能高低和规模大小分类。

按性能和规模分类

从20世纪60年代中期开始的按计算机性能高低和规模大小进行的分类,到80年代初最终形成了把计算机分为巨型计算机(包括小巨型计算机)、大型计算机、中型计算机、小型计算机(包括超级小型计算机)、工作站和个人计算机等几种类型的格局。

大型计算机(参见**大型计算机**)是通用系列计算机中的高端机种,它采用所在时期的最先进技术,性能高,容量大。在一个通用计算机系列中还有中档和低端机种,它们的性能和规模逐级降低,可以分别归入中型计算机和小型计算机类型。同一个系列中的各档计算机采用相同的操作系统,可以互相兼容(向上兼容和向后兼容)。

专门设计的小型计算机(参见**小型计算机**)也出现于20世纪60年代中期。它的字长短,容量小,初期产品的指令系统比较简单。在70年代先后出现了系列化的小型系列计算机和超级小型系列计算机。超级小型计算机(参见**超级小型计算机**)性能介于小型计算机和大中型计算机之间,字长一般为32位,有的达到64位。

巨型计算机(或称超级计算机)出现于20世纪70年代,它比大型计算机的速度更快,性能更高,是专为科学技术领域大量数值计算和数据处理的需要而研制的。最初有两种结构:单指令流多数据流(SIMD)阵列处理机(参见**阵列处理机**)和流水线向量计算机(参见**向量计算**)。到80年代又出现了多指令流多数据流(MIMD)大规模并行处理(MPP)系统(参见**大规模并行处理**)。80年代中期出现了小巨型计算机,它的价格与超级小型计算机相当,但性能接近巨型计算机。最初的小巨型计算机采用巨型计算机的高速流水线技术,但在指令系统、数据结构和存储管理等方面较多地吸收超级小型计算机的特点,使之有较高的性能价格比。其后又出现用微处理机并行处理系统的小巨型计算机。由于与相邻的类型在性能、规模和结构上的界限日趋模糊,小巨型计算机这一名词到90年代已不再使用。

自从20世纪70年代初出现微处理机之后,在计算机类型的最低端又添加了微型计算机。80年代初,微型计算机分为个人计算机和工作站两种类型。至此,在80年代形成了巨型计算机(包括小巨型计算机)、大型计算机、中型计算机、小型计算机(包括超级小型计算机)、工作站、个人计算机的分类格局。但随着微电子技术的发展,巨型计算机和大型计算机系统结构的先进技术不断下移到低端机中,而低端的机种,特别是工作站和个人计算机的性能更是不断提高,使上述分类中的各相邻类型间的界限越来越模糊。

进入90年代后,计算机的分类类型发生了变化。原来采用双极型电路的巨型计算机、大型计算机乃至小型计算机都先后采用CMOS微处理机组成的多处理机结构,原有的机种逐渐消失,新出现的一些计算机仅仅是在并行处理机的数量、采用的互连技术和微处理机型号方面有所不同。与此同时计算机在网络和通信中的地位日显突出,因而在商品市场上大量销售的计算机类型逐渐为服务器(包括超级服务器)、工作站、台式个人计算机和移动式计算机这样一些与网络和通信相关的机型所代替。这一时期,还把性能和规模处于最高端的计算机称为高性能计算机(参见**高性能计算**)。高性能计算机一般可以用来做成超级服务器,但某些性能极高和规模极大的高性能计算机往往专门用于解决某方面极为复杂的问题,例如日本地球模拟器中心的SX6向量计算机、美国的ASCI系统等。

服务器在网络环境或客户-服务器环境中为客

户提供服务的计算机。服务器的性能和规模涵盖了很大的范围,可把服务器由大到小分为企业级服务器、部门级服务器和群组级服务器。企业级服务器是服务器中的高端,相当于原来的大型计算机。每一种服务器系列都有良好的可伸缩性,在有的服务器系列产品中,并行工作的处理机可从4台一直扩充到几百台甚至于上千台,极高性能的服务器称为超级服务器。服务器的结构有分布式共享存储器的多处理机系统(参见**共享存储**)、松耦合的大规模并行处理(MPP)多计算机系统(参见**大规模并行处理**)或集群式计算系统(参见**集群式计算**)。很多服务器设计成专供某一方面的应用,例如Web服务器、邮件服务器、文件服务器,有的则设计成多功能服务器。可以支持科学与工程计算、数据处理、事务处理和网络信息服务等多方面的应用。

工作站属于台式计算机,其性能高于个人计算机,它可以在个人计算环境中或在分布式网络计算环境中工作。一般用2到4个微处理机组成,采用对称式共享存储器结构。

台式个人计算机一般是指单个用户独用的价格低廉的**微型计算机**,可以联网发送电子邮件和进行浏览等,它是用单个微处理机做成的。

移动式计算机发展迅速,已开发出笔记本电脑、手持计算机和可穿戴计算机等。移动式计算机一般可以通过无线传输成为移动的结点,实现移动网络计算。**笔记本电脑**可以随身携带,它开始出现于20世纪80年代末。手持计算机是一种**个人数字助理**,可以拿在手里使用。可穿戴计算机(参见**可穿戴计算**)的部件设计成一些可以合理分布于人体多个部位的模块,各个模块间的互连由近距离无线通信实现。它具有十分友好的人机交互能力,可以将人的双手从键盘和鼠标上解放出来。

从系统结构角度分类

在计算机学术界内,有人从计算机系统结构的角度对计算机进行分类。最为流行的是1966年M. J. Flynn提出的分类法,这是按指令流和数据流的多重性,把计算机划分为SISD、SIMD、MISD和MIMD 4种类型(参见**并行处理系统**)。在已有的并行处理系统中,除个别的属于SIMD外,其余的全部都属于MIMD。根据互连网络和通信机制的不同,MIMD可分为紧耦合的共享存储器多处理机系统和松耦合的多计算机系统两大类。共享存储器多处理机系统又可根据共享存储器的物理位置,分为集中式共享存储器多处理机系统和分布式共享存储器多处理机系

统。共享存储器多处理机还可根据每个处理机的作用是否完全相同分为对称式多处理机(SMP)系统和主从式多处理机系统两类。松耦合的多计算机系统可根据互联网络和通信机制的不同,分为消息传递多计算机系统(参见消息传递)和集群式计算机系统(参见集群式计算)。多处理机系统还可以按所用的处理机是否相同分为两类:同构型多处理机系统和异构型多处理机系统。

按其他方法分类

在数字计算机发展的过程中,曾以电子开关器件的更新作为计算机分类的特征,把电子管计算机称为第一代计算机,晶体管计算机称为第二代计算机,中小规模集成电路计算机称为第三代计算机,20世纪70年代中期以来采用大规模和超大规模集成电路的计算机统称为第四代计算机。

计算机还可以从应用范围的角度分为通用计算机和专用计算机。通用计算机可用于各种不同类型的应用,例如科学与工程计算、数据处理、事务处理和过程控制等。专用计算机只适合某一方面特殊的应用,例如专门用于控制生产过程的过程控制计算机(参见过程控制计算机)。随着微电子技术的发展,通用微处理机芯片的集成度和性能价格比不断提高。在很多场合下通用计算机已可涵盖和替代专用计算机。在另外很多场合下,专用计算机可以直接装入机电设备、仪器仪表或家电设备内部,成为其中的一个部件,这就是嵌入式计算机。

在计算机采用双极型线路的年代里,还有一种位片计算机。当时芯片的集成度低,有一处做法是把计算机的各种逻辑功能部件沿字长方向划分为一定宽度的标准模块。把这些模块级联起来,就可以组成任意宽度的数据通路,从而构成一台计算机,这就是位片计算机。进入90年代以后,位片计算机已经不再生产。还有的把中央处理机、存储器和输入输出接口集成在一个芯片上的微型计算机称为单片计算机或单片机,主要用于嵌入式控制领域。

对于在可靠性要求很高的场合下使用的计算机,如果计算机在出现故障时没有或几乎没有停顿时间,计算机仍能继续运行,这种计算机就是容错计算机(参见容错计算机)。如果出现故障时还需要有几秒钟、几分钟乃至更长的间断时间来采取自动弥补措施,则称为高可用计算机。还有为适应军事、航天、野外、海洋、工业等特定恶劣环境而设计的抗恶劣环境计算机(参见抗恶劣环境计算机),为满足环境保护要求而设计的绿色计算机(参见绿色计算

机)等。

还可按计算机对高级语言支持的程度,或者说按机器语言与高级语言的差距,把计算机分类为精简指令集计算机(参见精简指令集计算机)、复杂指令集计算机(参见复杂指令集计算机)以及和语言对应的系统结构的计算机和直接执行语言的系统结构的计算机。在和语言对应的系统结构的计算机中,机器语言与高级语言基本上是一一对应的,但需要经过软件或硬件翻译。直接执行语言的系统结构的计算机能直接执行高级语言原码,不需要经过中间翻译。

还可以按计算机驱动原理的不同把计算机分为控制驱动、数据驱动和需求驱动三种类型的计算机。控制驱动的计算机就是传统的以冯·诺依曼结构为基础的计算机,它执行操作的次序受指令计数器的控制。后两种是非传统计算机。数据驱动的计算机就是数据流计算机(参见数据流计算机),它的操作在全部操作数都到齐时即开始执行。需求驱动的计算机就是归约机,它的操作仅在需要用到这个操作的结果时才开始执行。

除了用微电子技术做成的集成电路组成计算机外,还在研究传统的电子器件以外的基本部件和工作原理。例如用有机或生物材料构成的分子器件(参见生物计算)、光器件(参见光计算机)和量子器件(参见量子计算),分别用它们组成具有特色的生物计算机、光计算机和量子计算机。(孙强南)

jisuanji liushuixian

计算机流水线 (computer pipeline) 把计算机的指令或费时的复杂操作分解成一系列可以独立执行的简单步,按流水线方式交迭进行的一种操作方式。每个步的执行部件称为站,每个站将本步的执行结果送往下一站后,又去执行下一条指令或下一个操作的这一步。这类似于工厂的生产流水线,工件放在传送带上,顺序地流过各个工位,每个工位在规定的时间内,完成本工位的装配任务。流水线的特点是在一项任务未完成前,可以启动新的任务,任务完成的速率与任务流过流水线的总时间无关,仅取决于任务送入流水线的速率,从而提高了整体处理速度。

计算机流水线的站由组合逻辑线路组成,完成特定的算术和逻辑操作。相邻的两个站由寄存器隔开,信息在站间的流动由时钟信号C来实现同步。图1是流水线基本结构的示意图。在图中,L表示

寄存器。

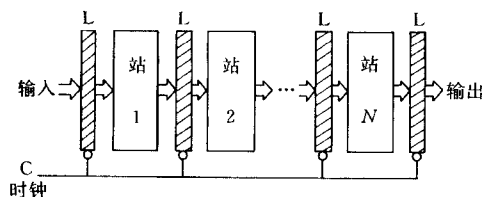


图1 流水线基本结构

计算机流水线的发展简史

将流水线概念用到计算机,可以追溯到1946年冯·诺依曼等人设计的第一台存储程序计算机。当时,提出了在运算器和输入输出设备之间设置一个缓冲器,使运算和输入输出操作可以同时进行。尽管冯·诺依曼在他的第一台计算机中并没有实现这种技术,但是,在后来的计算机中,这种运算和输入输出操作的重叠概念被广泛采用,并且成为计算机流水线的原始形式。

在晶体管计算机时代,计算机的主存储器采用磁心。由于磁心存储器的存取周期比晶体管中央处理器的周期长十倍左右,因此,中央处理器在执行指令的同时,可以启动1个或多个并发的存储器访问,这就构成了访存流水线。

计算机流水线设置多个站来提高处理速率。在20世纪60年代,逻辑元件的价格相对较高,流水线技术只用于价格昂贵的大型或巨型计算机。60年代初,IBM公司的STRETCH和CDC公司的6600计算机是典型的大量使用流水线技术的计算机系统,对后来大型和巨型计算机的体系结构产生了深刻的影响。随着微电子技术的迅猛发展,到80年代,逻辑元件的价格、功耗都大幅度下降,靠增加硬件来获得高速度的流水线技术在各种档次、各种类型的计算机中都得到了广泛应用。甚至在价格十分低廉的Intel 8086微处理器中,也能在执行运算的同时,使用流水线技术访问存储器。

计算机流水线工作原理

流水线的加速原理可用图2来说明。图2(a)表示串行处理过程,假如1个作业有 N 步,完成1步需要1个单位时间,则完成 N 步需要 N 个单位时间。图中的每一个方框表示执行1步,方框中的标号为执行的步号。图2(b)表示 M 个作业用流水线技术处理的情况。图中每一行方框表示1个作业随时间的变化,每一列方框表示在某个特定时刻,作业流在流水线中执行的步号。可看出,在图2(a)中,

完成1个作业需要 N 个周期,而在图2(b)中,如果 $M \gg N$,则完成1个作业只需约1个周期。

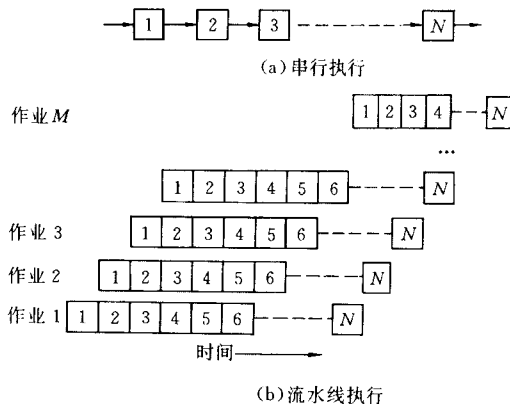


图2 流水线加速原理

计算机流水线的分类

计算机流水线一般可以分为指令流水线和运算流水线。

指令流水线 典型的指令执行过程可分解为若干步骤,这里假定为5步:①取指令;②指令译码;③形成操作数有效地址;④取操作数;⑤执行指令,含存操作结果和修改程序计数器。假如把这5个步骤分别安排在相互独立工作的站中进行,并且使每个站的执行时间大致相同,都是1个时间单位。这5个步骤的处理过程可按图2(b)实现指令流水线。

在一般情况下,流水线的第一站要连续不断地取指令,通常指令是顺序存放的,可以通过程序计数器加1来确定下一条指令的地址。但是,遇到条件转移指令,只有到流水线的最后一站才能知道下一条指令的地址。若在流水线的最后一站不对转移类指令采取特殊措施,在一条转移指令到达最后一站产生目的地址以前可能已经有若干条本不该执行的指令进入了流水线,改变了1个或多个机器寄存器的状态。为了保证正确执行,最简单的方法是使取指令和执行指令这两步进行互锁。在最后一站取出条件转移指令后,直到它到达最后一站前都不取新的指令,在最后一站产生正确的目的地址后才解除互锁,并使第一站用新的地址继续取指令。采用互锁时,流水线遇到转移指令就形成一段空闲状态。在大多数计算机中,条件转移出现的频率很高,平均5条~10条指令出现1次,简单的互锁将损失很多时间。

为了解决转移指令对流水线效率的影响,提出

了多种方案。对无条件转移指令提出了设置指令缓冲区的方法。在指令缓冲区中,预取足够多的指令,假如无条件转移后的地址落在指令缓冲区中,无条件转移可以不引起执行部件的停顿。对于条件转移指令,确定转移方向的条件码要在流水线后端的执行部件中产生,对流水线效率影响大。为了减少条件转移指令对流水线效率的影响,提出了多种方法(参见**转移目标缓冲器**),例如加快和提前形成条件码的方法;按一个方向猜测执行的方法;预取转移目标方法,即设置多套指令缓冲寄存器,发现条件转移指令后,同时按两个分支方向预取指令。此外,还有用编译程序自动调整条件转移指令位置的延迟转移方法等。

影响流水线效率和运行正确性的,除了转移指令外,还有数据相关。例如,在流水线中某条指令的结果可能被后面的指令使用,这时,后面的指令必须等前面指令的结果产生后才能执行。又如,流水线中某条指令的结果要存入某寄存器,但是该寄存器中原来的内容要由前面的指令来读取,那么,只有等前面指令读取该寄存器内容后,后面的指令才可写入该寄存器。再如,流水线中两条不同的指令有可能写同一地址,由于流水线会改变指令的执行次序,有可能造成该地址内容出错。

上述各种原因使指令流水线的设计变得错综复杂。在设计流水线时,必须确保计算机在执行转移指令和数据相关的情况下,仍能正确和高效地运行。

运算流水线 计算机的某些运算很费时间,例如浮点运算、乘法、除法等。为加快运算速度,这些运算也可以采用流水线技术。现以浮点加法为例加以说明。

假设输入的是一对规格化的浮点数,分4步完成浮点加法:①对阶,找出阶码较大的数,求阶差,较小的数尾数右移;②尾数相加,产生和的尾数;③规格化和的尾数,调整和的阶码;④尾数舍入,如果舍入引起尾数溢出,再规格化和调整阶码。

上述4步可以构成1条由4个站组成的浮点加法流水线。

运算流水线又可以细分为单功能和多功能流水线、静态和动态流水线以及标量和向量流水线3类。

计算机流水线技术的发展趋势

随着微电子技术和计算技术的发展,流水线技术已应用到从微型计算机到巨型计算机的各种类型的计算机中。值得一提的是,流水线技术已成为实现**精简指令集计算机(RISC)**的首选技术。在RISC

中,最初的目标是每个时钟周期执行1条指令,后来提出了在1个时钟周期内执行多条指令的目标,于是便出现了超标量、超流水线和超长指令字的体系结构(参见**多发射结构**)。在每种体系结构中,出现了多条流水线或多个功能部件,使流水线技术进入了更高的发展阶段。

参考文献

1. Stone H S. High-Performance Computer Architecture. 3rd Ed., Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1993
2. Hwang K, Briggs F A. Computer Architecture and Parallel Processing, New York: McGraw-Hill, 1984
3. Hennessy J L, Patterson D A. Computer Architecture: A Quantitative Approach. San Mateo, Calif: Morgan Kaufmann, 1990 (韩承德)

jisuanji ruanjian

计算机软件(computer software) 计算机系统程序及其文档。程序是计算任务的处理对象和处理规则的描述;文档是为了便于了解程序所需的阐明性资料。程序必须装入机器内部才能工作,文档一般是给人看的,不一定装入机器。

细言之,软件一词具有三层含义。一为个体含义,即指计算机系统程序及其文档;二为整体含义,即指在特定计算机系统中所有上述个体含义下的软件的总体。三为学科含义,即指在研究、开发、维护以及使用前述含义下的软件所涉及的理论、原则、方法、技术所构成的学科。在这种含义下,软件宜称为软件学,但一般仍称作软件。

软件一词源于程序,到了20世纪60年代初期,人们逐渐认识到和程序有关的文档的重要性,从而出现了软件一词。

软件是用户与硬件之间的接口界面。要使用计算机,就必须编制程序,必须有软件。用户主要是通过软件与计算机进行交往。软件是计算机系统的重要依据。为了方便用户,为了使计算机系统具有较高的总体效用,在设计计算机系统时,必须通盘考虑软件与硬件的结合,以及用户的要求和软件的要求。软件在计算机系统中起指挥、管理作用。计算机系统工作与否,做什么以及如何做,都是听命于软件。

发展计算机科学技术,软件和硬件都是不可缺少的重要方面。二者既有分工,又有配合。软件的

发展以硬件为基础,其发展也促进了硬件、计算机科学技术、以及其他科学技术的发展。它在社会信息化和人类文化的发展中具有重要作用。

发展过程

软件的发展受到应用和硬件发展的推动和制约,其发展过程大致可分为三个阶段。

从第一台计算机上的第一个程序的出现到实用的高级程序设计语言出现以前为第一阶段(1946年—1956年)。计算机的工作是由储存在其内部的程序指挥的。这是冯·诺依曼式计算机的重要特色。当时计算机的应用领域较窄,主要是科学计算。就一项计算任务而言,输入、输出量并不大,但计算量却较大,主要处理一些数值数据。机器结构以中央处理器为中心,存储容量较小。编制程序(简称编程)所用的工具是低级语言,即以机器基本指令集为主的机器语言和在机器语言基础上稍加符号化的汇编语言。突出的问题是,程序的设计和编制工作复杂、繁琐、费时和易出差错。衡量程序质量的标准主要是功效,即运行时间省、占用内存小,很少考虑到结构清晰、易读性和易维护性。设计和编制程序采用个体工作方式,强调编程技巧,主要研究科学计算程序、服务性程序和程序库,研究对象是顺序程序。当时人们对和程序有关的文档的重要性尚认识不足,重点考虑程序本身,尚未出现软件一词,但毕竟由于程序是软件的主体,从发展的连续性来看,故将其归为第一阶段。

从实用的高级程序设计语言出现以后到软件工程出现以前为第二阶段(1956年—1968年)。随着计算机应用领域的逐步扩大,除了科学计算继续发展以外,出现了大量的数据处理问题,其性质和科学计算有明显区别。就一项计算任务而言,计算量不大,但输入、输出量却较大。这时,机器结构转向以存储控制为中心。出现了大容量的存储器,外围设备发展迅速。为了提高程序人员的工作效率,出现了实用的高级程序设计语言。为了充分利用系统资源,出现了操作系统。为了适应大量数据处理问题的需要,开始出现数据库及其管理系统。在20世纪50年代后期,人们逐渐认识到和程序有关的文档的重要性,从而到了60年代初期,出现了软件一词,融程序及其有关的文档为一体。这时,软件的复杂程度迅速提高,研制周期变长,正确性难以保证,可靠性问题相当突出。到了60年代中期,出现了人们难以控制的局面,即所谓软件危机。为了克服这一危机,人们进行了以下三个方面的工作:①提出结构

程序设计方法;②提出用工程方法开发软件;③从理论上探讨程序正确性和软件可靠性问题。这一阶段的研究对象增加了并发程序,着重研究高级程序设计语言、编译程序、操作系统、以及各种应用软件。计算机系统的处理能力得到加强。设计与编制程序的工作方式逐步转向合作方式。

软件工程出现以后迄今为第三阶段(1968年以来)。由于大型软件的开发是一项工程性任务,采用个体或合作方式不仅效率低,产品可靠性差,而且很难完成,只有采用工程方法才能适应。从而在1968年的大西洋公约学术会议上提出了软件工程。三十多年来,软件领域工作的主要特点是,第一,随着应用领域的不断拓广,出现了嵌入式应用,其特点是受制于它所嵌入的宿主系统,而不只是受制于其功能要求。为了适应计算机网络的需要,出现了网络软件。随着微型计算机的出现,分布式应用和分布式软件得到发展;第二,开发方式逐步由个体合作方式转向工程方式,软件工程发展迅速,特别是,出现了“计算机辅助软件工程”。除了开发各类工具与环境,用以支撑软件的开发与维护外,还出现了一些实验性的软件自动化系统;第三,致力研究软件开发过程本身,研究各种软件开发风范与模型,例如,功能分解风范与模型,面向对象风范与模型。研究软件体系结构、基于构件的软件,以及中间件等;第四,除了软件传统技术继续发展外,人们着重研究以智能化、自动化、集成化、并行化以及自然化等为标志的软件开发新技术;第五,注意研究软件理论,特别是软件开发过程的本质。

分 类

软件一般可分为系统软件、支撑软件、应用软件三类。

系统软件 居于计算机系统中最近硬件的一层。其他软件一般都通过系统软件发挥作用。它与具体的应用领域无关,如编译程序和操作系统等。编译程序把程序人员用高级语言书写的程序翻译成与之等价的、可执行的低级语言程序;操作系统则负责管理系统的各种资源、控制程序的执行。在任何计算机系统的设计中,系统软件都要予以优先考虑。

支撑软件 支撑软件的开发、维护与运行的软件。随着计算机科学技术的发展,软件的开发、维护与运行的代价在整个计算机系统中所占的比重很大,远远超过硬件。因此,支撑软件的研究具有重要意义,直接促进软件的发展。当然,数据库管理系统、网络软件等也可算作支撑软件。但是20世纪

70 年代中后期发展起来的软件开发环境以及后来开发的中间件则可看成现代支撑软件的代表,软件开发环境主要包括环境数据库、各种接口软件和工具组。三者形成整体,协同支撑软件的开发与维护。

应用软件 特定应用领域专用的软件。例如,人口普查用的软件就是一种应用软件。对于具体的应用领域,应用软件的质量往往成为影响实际效果的决定性因素。70 年代出现的嵌入式应用,其相应软件的复杂程度高,开发工作量大,促进了软件的发展。模拟应用导致模拟语言(SIMULA)的出现。应用软件的作用越来越大。

基本内容

主要有**软件语言**、**软件方法学**、**软件工程**、以及**软件系统**等。

软件语言 用以书写软件的语言。可分为需求级语言、功能级语言、设计级语言、实现级语言,以及文档语言。需求级语言用以书写软件需求定义,又称需求定义语言,目前多数仍是非形式的语言或半形式的语言;功能级语言用以书写软件功能规约,又称功能规约语言,可以是形式的,也可以是非形式的;设计级语言用以书写软件设计规约,又称设计规约语言,一般是形式的;实现级语言用以书写实现算法,相对说来,较为成熟。低级语言、古典高级语言从 FORTRAN, ALGOL, COBOL, PASCAL, C, 直到 Ada 均属此类;文档语言用以书写文档,目前一般是非形式的。

软件方法学 软件开发全过程的指导原则与方法体系。其另一种含义是以软件方法为研究对象的学科。从开发风范上看,软件方法有自顶向下的开发方法、自底向上的开发方法。在实际软件开发中,大都是自顶向下与自底向上两种方法的结合,只不过是以前者为主而已。从性质上看,有形式方法与非形式方法。形式方法是一种具有坚实数学基础的方法,从而允许对系统和开发过程作严格处理和论证。非形式方法则不把严格性作为其主要着眼点。从适用范围来看,有整体性方法与局部性方法,适用于软件开发全过程的是整体性方法,自顶向下方法、自底向上方法、各种软件自动化方法等均属整体性方法。适用于开发过程个别阶段的为局部性方法,如适用于需求分析阶段的各种需求分析方法,适用于设计阶段的各种设计方法等。此外,由于程序设计方法的发展相对较为成熟,从而早在软件方法学出现以前,就出现了程序设计方法学,它研究各类程序设计方法,如过程式程序设计、逻辑式程序设计、

函数式程序设计、对象式(面向对象)程序设计、以及顺序程序设计、并发程序设计、并行程序设计、分布程序设计、可视程序设计、文化程序设计等等。

软件工程 应用计算机科学、数学及管理科学等原理,以工程化方法制作软件的工程,它是一门交叉性学科。软件工程的架构可用一三元组刻画,即 $SE = (G, P, Q)$, 其中 SE 表示软件工程, G 为目标, P 为原则, Q 为过程。目标 G 主要包括正确性、易用性以及价格合宜。正确性反映软件产品实现相应功能规约的程度, E. W. Dijkstra 曾说,“迄今的软件排错设施只能发现软件错误,不能断定软件没有错误”。20 世纪 60 年代中期以来,虽有不少计算机科学家致力于正确性问题的研究,但迄今为止,不论理论上还是实践上正确性问题均未得到很好解决。易用性反映软件的基本结构、实现及其文档为用户易用的程度。由于软件系统的用户广泛,要求因人而异,因而,易用性往往是软件工程难以满足的目标。价格合宜反映软件开发与运行的总代价满足用户要求的程度。随着硬件价格的日趋低廉,传统的系统性能显得不太重要,价格合宜扩大成包括软件运行与维护方面的价格考虑。由于复杂软件系统不可能一步开发成功,通常采用增殖式开发策略,先生产生出一个基本核心产品,随后再改进提高,以实现所需的全部功能。总体价格合宜便要求产品能便于修改与改进。原则 P 主要包括易变性原则、综合性原则、支撑性原则、以及管理性原则等,其中易变性原则指必须认识软件需求的易变性;综合性原则指综合使用各种方法与工具以实现软件目标;支撑性指应充分认识支撑软件的作用;管理性指必须认识管理过程的作用。过程 Q 涉及软件生存周期中的各类过程,主要包括基本过程、支撑过程以及组织过程。

软件系统 计算机系统中由软件组成的子系统,其组成成分可有系统软件、支撑软件以及应用软件。系统软件与特定解题领域无关,它主要包括操作系统、语言处理系统、数据库系统、分布式软件系统、网络软件系统,以及人机交互软件系统等。支撑软件主要包括用以支撑软件开发、维护与运行的各类软件开发环境和中间件系统。应用软件是特定应用领域专用的软件,种类繁多,不拟枚举。

难点与展望

软件是智力产品,软件开发是智力活动。其难点是,第一,对软件开发过程本质的认识,“变是不变的真理”在软件领域表现突出。软件在使用过程中必须不断维护,其中包括:①纠错性维护,使用中

一旦发现错误,便及时纠正;②适应性维护,随着环境的改变,使原有软件适应新的环境;③完善性维护,原有软件在设计时难免有不完善之处,在使用中视需要逐步完善。变动性是软件的重要特征。软件开发风范与模型均和领域有关。领域不同,合适的风范与模型可能也不同。但软件开发过程的本质是否也和领域有关,目前尚难断定。共性为何,个性为何,这些都是有待研究的问题。软件主要是由人开发的。各人思考问题的习惯、方式不尽相同,思维规律也不尽相同,但软件开发过程的本质是否也和思维规律有关,亦亟待研究。此外,软件开发过程的复杂性研究对软件开发的作用巨大,影响深远。第二,功能智能化。“智能”含义,众说纷纭。然而,多数人认为,智能指学习能力。如果软件智能化指的是使软件本身具备智能,则事实表明,软件是能智能化的。如果软件智能化指的是使软件具备人类智能,则当然是有条件的。目前流行的归纳学习、分析学习、联接学习以及遗传学习等途径各有利弊,应视具体情况,酌情采用,或另创新途径。第三,程序正确性。程序正确性总是相对某一基准而言的。一般说来,程序(实现级)是正确的,指的是它能全部体现相应设计规约中的功能,而设计规约的正确性,指的是它能体现功能规约中的全部功能,而功能级程序(即功能规约)是正确的,则指它能体现需求定义中的全部需求。过去考虑较多的是实现级程序的正确性,今后宜更多考虑设计级与功能级程序的正确性。理论上的正确性与实际的正确性从理论上说应该是一致的,但实际上又往往不能完全一致。问题在哪里,是理论原因还是实际原因,有待研究。此外,开发过程不正确,则任何级别的程序正确性均无法保证。在这种意义上说,开发过程的正确性就显得格外重要。第四,用户友善性。软件质量优劣的最终评判者是用户。对用户不友善的软件,用户当然不会乐于使用。为使软件对用户友善,人与系统的接口界面应尽可能采用自然语言与图形语言,这一点颇为重要,采用自然语言与图形语言还只是接口界面的表示问题,更重要的还是接口界面的结构。结构宜简明,力戒繁琐。

参考文献

1. 徐家福. 软件技术漫谈. 计算机科学, 1992, 19(1)
2. Freeman P. Software Perspectives: The System is the Message, Reading, MA: Addison-Wesley, 1987
(徐家福)

jisuanji ruanjian de falü baohu

计算机软件的法律保护 (legal protection of computer software) 以法律手段对计算机软件的知识产权提供保护和为支持计算机软件的安全运行而提供的法律保护。

计算机软件知识产权是指公民或法人对自己在计算机软件开发过程中创造出来的智力成果所享有的专有权利。包括著作权、专利权、商标权和制止不正当竞争的权利等。

对计算机软件知识产权加以保护是为保护智力成果创造者的合理权益,以维护社会的公正,维护软件开发成果不应无偿占用的原则,鼓励软件开发者的积极性,推动计算机软件产业及整个社会经济发展的尽快发展。

软件的权利人可拥有以下三方面知识产权:

该软件的表达(例如程序的代码、文档等)方面的权利——著作权;

该软件的技术设计(例如程序的设计方案、处理问题的方法、各项有关的技术信息等)方面的权利——专利权和制止不正当竞争的权利;

该软件的名称标识方面的权利——商标权。

计算机软件的著作权 著作权又称为版权,是指作品作者根据国家著作权法对自己创作的作品表达所享有的专有权利的总和。1990年9月我国颁布的《著作权法》规定,计算机软件是受著作权法保护的一类作品。1991年6月颁布的《计算机软件保护条例》作为著作权法的配套法规是保护计算机软件著作权的具体实施办法。我国的法律和有关国际公约认为:计算机程序和相关文档、程序的源代码和目标代码都是受著作权法保护的作品。

按照法律规定,软件开发者在一定的期限内对自己软件的表达(例如程序的代码、文档等)享有的专有权利包括发表权、开发者身份权、以复制、展示、发行、修改、翻译、注释等方式使用其软件的使用权、使用许可权和获得报酬权以及转让权。国家依法保护软件开发者的这些专有权利。对软件权利人利益的最主要的威胁是擅自复制程序代码和擅自销售程序代码的复制品,这是侵害软件权利人的著作权的行为。因此,软件的著作权是软件权利人的最主要的权利。

著作权法的原理是保护作品的表达,即作品本身,著作权法不保护作品的构思。对软件的著作权保护不能扩大到开发软件所用的思想、概念、发现、原理、算法、处理过程和运行方法。因此,参照他人

程序的技术设计,独立地编写出表达不同的程序的做法并不违犯著作权法。不过,对软件进行修改属于软件著作权人的专有权利。如果有人在他人程序著作权有效期内,擅自对他人程序进行修改改编,所产生的程序并没有改变他人程序设计构思的基本表达,在整体上与他人程序相似,则虽然在代码文字表达方面存在不同,仍属于侵害他人程序著作权的行为。

与计算机软件相关的发明的专利权 专利权是由国家专利主管机关根据国家颁布的专利法授予专利申请者或其权利继受者在一定的期限内实施其发明以及授权他人实施其发明的专有权利。这里所说的发明包括对产品的发明及其改进,也包括对方法的发明及其改进。世界各国用来保护专利权的法律是专利法,专利法所保护的是已经获得了专利权的、可以在生产建设过程中实现的技术方案。各国专利法普遍规定,能够获得专利权的发明应当具备新颖性、创造性和实用性。中国的《专利法》已经于1984年3月颁布。

一般地说,计算机程序代码本身并不是可以申请发明专利的主题,而是著作权法的保护对象。不过,同设备仪器结合在一起的计算机程序可以作为一项产品发明的组成部分,同整个产品一起申请专利。此外,一项计算机程序无论是否同设备仪器结合在一起,如果在其处理问题的技术设计中具有发明创造,在不少国家里,这些与计算机软件相关的发明创造可以作为方法发明申请专利,很多有关地址定位、虚拟存储、文件管理、信息检索、程序编译、多重窗口、图像处理、数据压缩、多道运行控制、自然语言翻译、程序编写自动化等方面的发明创造已经获得了专利权。在我国,不少有关将汉字输入计算机的发明创造也已经获得了专利权。一旦这种发明创造如果获得了国家专利主管机关授予的专利权,在该专利权有效期内,其他人在开发计算机程序时就不能擅自实施这种发明创造,否则将构成侵害他人专利权的行为。

有关计算机中商业秘密的不正当竞争行为的制止权 如果一项软件的技术设计没有获得专利权,而且尚未公开,这种技术设计就是非专利的技术秘密,可以作为软件开发者的商业秘密(Trade secret,也有人译为工商秘密或者营业秘密)而受到保护。把软件的尚未公开的技术设计作为商业秘密保护,是保护软件技术设计的主要途径。一项软件的尚未公开的源程序清单通常被认为是开发者的商业秘

密。有关一项软件的尚未公开的设计开发信息,如需求规格、开发计划、整体方案、算法模型、组织结构、处理流程、测试结果等都可被认为是开发者的商业秘密。

对于商业秘密,其拥有者具有使用权和转让权,并可以许可他人使用,也可以将之向社会公开或者去申请专利。不过,对商业秘密的这些权利不是排他性的。任何人都可以对他人的商业秘密进行独立的研究开发,也可以采用反向工程方法或者通过拥有者自己的泄密行为来掌握它,并且在掌握之后使用、转让、许可他人使用、公开这些秘密或者对这些秘密申请专利。然而,根据我国1993年9月颁布的《反不正当竞争法》,商业秘密的拥有者有权制止他人对自己商业秘密从事不正当竞争行为,这里所称的不正当竞争行为包括:以不正当手段获取他人的商业秘密,使用以不正当手段获取到的他人的商业秘密,接受他人传授或透露了商业秘密的人(例如商业秘密拥有者的职工、合作者或经商业秘密拥有者许可使用的人)违反事前约定,滥用或者泄露这些秘密。

一项信息成为商业秘密的前提在于其本身是秘密。一项商业秘密一旦被公开就不再是商业秘密。为了保护商业秘密,最基本的手段就是依靠保密机制,包括在企业内建立保密制度、同需要接触商业秘密的人员签订保密协议等。

计算机软件名称标识的商标权 对商标的专用权也是软件权利人的一项知识产权。所谓商标是指商品的生产者或者经销者为使自己的商品同其他人的商品相互区别而置于商品表面或者商品包装上的标志,通常用文字、图形或者兼用这两者组成。

国际软件行业现在十分重视商标的使用。有些商标用于标识提供软件产品的企业,如“IBM”,“DEC”,“MS”等,它们是对应企业的信誉的标志。有些商标则用于标识特定的软件产品,如“UNIX”,“OS/2”,“Lotus 1-2-3”等,它们也是特定软件产品的名称,是特定软件产品的功能和性能的标志。

在一般情况下,一个企业的标识或者一项软件的名称未必就是商标。然而,当这种标识或者名称在商标管理机关获准注册、成为商标后,在商标的有效期内,注册者对它享有专用权,他人未经注册者许可不得再使用它作为其他软件的名称。否则,就构成冒用他人商标、欺骗用户的行为。很多国家颁布了商标法以保护商标注册者的这种专用权,我国的《商标法》已经在1982年8月颁布。(应明)

jisuanji shijue

计算机视觉 (computer vision) 对描述景物的一幅或多幅图像的数据经计算机处理,以实现类似于人的视觉感知功能,是人工智能的重要组成部分。

有些学者把为实现视觉感知所需要进行的图像获取、表示、处理和分析等也包含在计算机视觉中,使整个计算机视觉系统成为一个能够看的机器,从而可以对周围的景物提取各种有关信息,包括物体的形状、类别、位置以及物理特性等,以实现对物体的识别理解和定位,并在此基础上作出相应的决策。

景物在成像过程中经透视投影而成光学图像,再经过取样和量化,得到由各像元的灰度值组成的二维阵列,即数字图像,这是计算机视觉研究中最常用的一类图像。此外,还用由激光或超声测距装置获取的距离图像,它直接表示物体表面一组离散点的深度信息。用多种传感器实现数据融合则是近年来获取视觉信息的重要方法。

人的视觉能力很强,似乎毫不费力就可以对周围的景物一目了然。但要使计算机完全具有或接近人的视觉能力,却是一个遥远的目标,迄今为止,人类对自己的视觉感知机制还不十分了解。因此,计算机视觉作为一个学科分支,其中心任务是通过图像或图像序列的分析,得到景物的尽可能完全和正确的描述。研究完成这一任务的理论、方法及其软硬件实现,称为图像理解,许多人把它看成是计算机视觉的同义词。由于物体的几何形状、物理性质、摄像机特性、照明情况以及物体与摄像机之间的空间关系等诸多因素在成像过程中被综合,仅仅表现为图像中像元的灰度值,以解决其反问题为目标的图像理解或计算机视觉,无疑是十分困难的,它的研究涉及到神经科学、应用数学、图像处理、模式识别、知识工程等多个学科分支的相互交叉和渗透。但从应用的观点来看,却有许多办法可以由图像中提取信息来解决与景物有关的实际问题,而不需要完全确定景物的三维结构。研究这种面向应用的计算机视觉系统的设计与实现技术,称为机器视觉。

计算机视觉的研究已有三十多年历史。早期研究工作集中在二维的情形,例如字符识别、染色体显微图像分类等。1965年, L. G. Robert 设计和实现了一个理解多面体积木世界景物的程序,是计算机三维视觉的最早成果。在该程序中,建立了一个含有立方体、长方体、楔块和六棱柱的三维物体模型库。由线画图识别这些物体的方法是先根据顶点数目之

类的简单特征确定候选的三维模型,再经过平移、旋转、尺度变换和投影,与待识线画图进行匹配。70年代中,计算机视觉在遥感图片解释、生物医学图像分析、工业自动检验等方面都取得了不同程度的成功。随着传感技术的发展,开始了距离图像获取和分析的研究,如由移动传感器对静止景物或静止传感器对运动物体获取的图像序列的分析,已成为受到重视的新方向;这种通过对运动的分析,有可能确定物体的三维形状和运动参数,可用于移动机器人导航或工业机器人视觉系统。1978年, H. G. Barrow 和 J. M. Tenenbaum 提出可以通过产生各种固有的景物描述性表示如深度信息、表面朝向等来改进机器视觉系统的处理能力。80年代初期 D. Marr 在此基础上建立了比较系统和完整的视觉计算理论。Marr 理论开创了计算机视觉蓬勃发展的新局面,并主导了整个80年代的研究工作。直到1990年, J. Aloimonos 提出有目的的、定性的主动视觉等新理论框架,才形成当今计算机视觉研究的新热点。

计算机视觉方法的研究经历了几个发展阶段。在80年代以前,多数研究工作都是针对个别的有限制的问题进行的,其基本方法是:①获取灰度图像;②从图像中提取边缘、周长、惯性矩等等特征;③从描述已知物体的特征库中选择特征匹配最好的相应结果。由于缺乏三维信息,上述方法很难在复杂的环节中取得较好的效果。80年代是计算机视觉方法的一个新发展阶段,按照 Marr 理论(参见视觉计算理论),视觉是通过自下而上的三个层次的信息处理过程来实现的,即①对图像进行边缘检测与图像分割等低层视觉处理(参见图像分析);②求取深度信息、表面朝向等 $2\frac{1}{2}$ 维描述,主要方法有由影调、轮廓、纹理等恢复三维形态(参见三维形态恢复),由体视恢复景物的深度信息(参见立体视觉),由图像序列分析确定物体的三维形状和运动参数(参见运动分析),距离图像获取与分析以及结构光方法等(参见计算机视觉中的结构光法);③根据三维信息对物体进行建模、表示与识别,可采用基于广义圆柱体的方法(参见视觉计算理论),另一常用方法是将物体外形表示为平面或曲面块(简称面基元)的集合,每个面基元的参数以及面基元之间的相互关系用属性关系结构来表示,从而将物体识别问题转化为属性关系结构的匹配问题。到80年代末,尽管在上述三个层次信息处理模块的计算理论、算法与表达乃至硬件实施都有很大进展,仍然不能

实现集成起来的通用计算机视觉系统。于是,一些学者开始寻找突破 Marr 理论框架的新途径。以 1990 年 J. Aloimonos 提出定性视觉、主动视觉等为标志,计算机视觉方法发展到当前的新阶段。定性视觉方法的核心是将视觉系统看成执行某一任务的更大系统的子系统,视觉系统所要获取的信息,只是完成大系统任务所必需的信息。在许多情况下,只要景物的定性或不完全描述就够了,无需如 Marr 理论所要求的完整的定量描述。景物的描述与识别,也不只经历一个自下而上的单向信息处理过程,而是在自下而上与自上而下相结合的控制策略下实现的。主动视觉方法则集感知、规划与控制为一体,通过这些模块的动态调用和信息获取过程与处理过程的相互作用,来更有效地完成视觉任务。该方法的核心是主动感知机制的建立,就是根据当前任务、环境状况、阶段处理结果和有关知识,来规划和控制下一步获取信息的传感器类型及其位姿。实现多视点或多传感器的数据融合,也是其关键技术。

计算机视觉的应用范围很广,许多二维视觉系统已经商品化,在实际生产和生活中发挥着重要作用。例如,早已为超级市场商品自动计价广泛采用的条形码识别系统;公安侦破和保安应用的指纹自动鉴定系统;信函分拣和办公自动化应用的文字识别系统等。还有一些生物医学图像分析和遥感图片自动解释的系统也达到了实用化程度。在工业自动检验方面,已经实用的有无损探伤系统。例如,1973 年日立公司研制的印刷电路板自动检验系统,可在 1/60 s 内完成一个视场的检验,一块电路板的检验依其尺寸大小只需 0.17 ~ 1.7 s。1976 年日本研制成功全自动电路焊接系统,这是机器视觉用于工业生产的较早例子,该系统每小时可组装 2 000 块晶体管芯片。基于计算机层析摄影的器官图像三维重构和显示,是计算机视觉在辅助医学诊断方面的成功应用。计算机视觉还曾用于在海湾战争中使用过的战斧式巡航导弹的制导。该视觉系统具有近红外和可见光的传感器及数字场景面积匹配器,在距目标 15 km 的范围内发挥作用。机器人也是计算机视觉应用的一个重要领域,对于无人驾驶自主车的自动导航,以及在工业装配、太空、深海或危险环境(如核辐射)中代替人工作的自主式机器人,计算机三维视觉是不可缺少的一项关键技术。80 年代后期以来,这方面已有一些实验系统陆续研制成功。例如,由美国国家宇航局提出,并由 Carnegie-Mellon 大学机器人研究所研制的六腿漫步机器人 Ambler,

具有自动行走和采集样本的能力,以在外星球表面观察和收集有关物理、气象和生物状况的资料为目的。按照观察前方、规划行进道路和挑选样本的不同要求,其视觉系统配备有立体视觉用的彩色摄像机和获取距离图像用的激光测距仪,具有对崎岖地形和障碍物进行大范围三维描述以及对附近环境进行细致理解的功能。再如,美国 Martin Marietta 公司研制的自主车,在视觉引导下可行驶于一般道路,最高速度达 20 km/h,并具有一定的避障能力。美国麻省理工学院(MIT)研制的 Polly 移动机器人,其视觉系统可完成室内环境下的道路识别和避障等。

计算机视觉的发展,提高了自动化和机器智能水平,为发展智能机器人和各种智能系统,提供了一项关键技术。随着主动视觉、定性视觉、多传感器信息融合等新方向的深入研究,以及更多地面向应用,可以预料,计算机视觉将有更诱人的发展。

参考文献

1. Marr D 著. 视觉计算理论. 姚国正等译. 北京: 科学出版社, 1987
2. 李介谷. 计算机视觉的理论和实践. 上海: 上海交通大学出版社, 1991
3. Aloimonos J. Purposive and Qualitative Action Vision. Proc. of 10th ICPR. 1990 (石青云)

jisuanji shijue zhong de jiegou guangfa
计算机视觉中的结构光法 (structured light method in computer vision) 利用具有一定光栅几何结构形状的光源投射到景物上,通过计算光栅变形获取三维景物信息的方法。投射光栅的结构形式可以是斑点、线条、网格、圆等等。

三维景物信息的获取是计算机视觉中的主要问题。传统的方法是利用成像设备获取景物的灰度图像,从灰度图像中获取三维信息。由于图像中各灰度值包含了物体表面的几何形状、反射特性、纹理、照度及成像设备特性等大量间接信息,使得从灰度图像中提取景物的三维信息,进行景物的识别与理解变得非常困难。人们开始试图研究获取景物的三维空间信息的直接方法,结构光法就是其中的一种。1971 年 P. M. Will 和 K. S. Pennington 将光栅投射到景物上计算景物的三维信息,获得景物的深度图像(或称距离图像),成为使用结构光法的最早的实例。使用深度图像与使用灰度图像相比可以更为便利地解决景物理解和识别中的有关问题,因此引起人们极大的关注。结构光法由于其结构简单,可以

避免复杂的对应点匹配问题,更是成为研究的热点,研究成果不断涌现。已经出现了利用结构光法进行三维信息获取的比较成熟的商品化系统,结构光法成为在特定应用中普遍使用的一种三维信息获取方法。

结构光法的基本思想是利用投射光中的几何结构信息帮助提取景物的三维几何信息。下面以最常用的投射光为平面光面的例子说明结构光法的基本原理。从图1可见,当由光源发出的平面光面投射到景物上时,在景物上就出现一明亮的光条,由于其他部分比较暗淡,只有这明亮光条部分才能被拍摄和被检测到,并在成像系统中成像。成像系统中每个成像点和成像系统的焦点确定了一条三维空间中的视线,而产生这个像点的景物点亦必然在此视线上。此外,这个景物点也保证是位于三维空间的光面上,而光面与视线仅相交于一点。由于光面的方向及位置可以是预先设置的,是已知的,因而通过计算视线与光面的交点就可以得到景物点的三维空间的位置。实用中,光面可以由激光或一条狭长缝隙的投射光来产生。成像设备应当只能检测到投射产生的光条,这就意味着成像是在暗室中进行的。但是,如果成像设备装有适当的滤光镜,以激光作为投射光的系统就能在正常光线下工作。而且激光的另一个优点是它能聚焦成非常薄的光面,因此在结构光法中把激光作为投射光源最为常见。

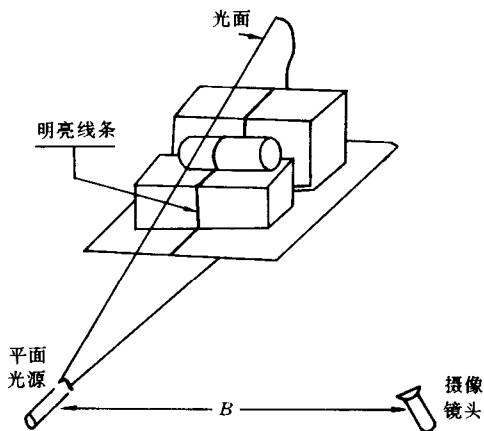


图1 结构光法原理图

参考文献

Fu K S, Gonzalez R C, Lee C S G. Control, Sensing, Vision and Intelligence. McGraw-Hill Book Company, 1987

(袁保宗)

jisuanji shuxue

计算机数学 (computer mathematics) 研究用计算机进行数学问题符号演算和推理的学科。计算机数学是随着计算机科学和数学的发展而逐步融合形成的。由于计算机科学和计算机的出现与发展,使得数学问题的机械计算和证明成为实际可能,并且反过来大大推动了计算机科学和数学的发展。

计算机数学和作为数学一个分支的计算数学既有区别又有联系。历史上,计算数学是以数学分析和代数学为基础发展起来的近似计算方法,与计算机数学有着不同的起源背景,它的输入和输出都是数值,在研究问题的目的、方法和技巧方面都形成了自己的体系。而计算机数学的输入和输出都是数学表达式,所涉及的是符号演算。例如计算方程

$$x^3 + 3x^2 + 3x + 2 = 0$$

的解,计算机数学的方法给出精确解(解析解)

$$x_1 = -2, \quad x_2 = \frac{-1 + \sqrt{-3}}{2}, \quad x_3 = \frac{-1 - \sqrt{-3}}{2},$$

而数值计算的方法给出近似解(数值解)

$$x_1 = -2, \quad x_2 = -0.5 - 0.866025i,$$

$$x_3 = -0.5 + 0.866025i$$

数学中的很多问题都属于非数值计算,例如计算不定积分,计算一个理想的准素分解,推导数学定理等,在这个领域,计算机数学有很广阔的理论和应用发展前景,即使在传统的数值计算工作中,有时也需要借助计算机数学的手段,以丰富和改善数值计算的结果。计算机数学的内容涵盖了数学的许多方面,例如定理机器证明(自动推理)、计算机代数、计算几何等,并伴随日益增多的应用。使用计算机数学辅助设计机械结构,不仅可以完整地描述机构的运动状态,而且可以发现一些极端条件下隐含的缺陷,大大弥补了传统设计方法的不足。使用计算机数学开发的机器人运动控制程序,可以从运动学和动力学的角度设计出最佳的运动路线,避开障碍物。

用计算机解决数学问题,关键是设计相关的算法,从这一点上说,计算机数学就是研究各种数学问题的算法。尽管计算机本质上只能处理有限的对象,在一定意义上,计算机数学也能驾驭无穷对象,处理涉及到连续对象的一些数学问题。

数学的进步总是伴随着计算方法的完善和证明能力的提高。但只是在计算机问世以后,才真正刺激了数学问题算法设计的快速发展。1958年,王浩设计了一个计算机程序,在9分钟的时间里,证明了

罗素和怀德海《数学原理》中几百条定理。并提出了“数学机械化”的概念,这是计算机数学作为一个独立研究领域的开端。从此以后,计算机数学在数学的各个领域迅速渗透,随着计算机科学和技术的发展,一些阻碍计算机数学应用的问题被逐渐克服,例如运算的速度、存储器的容量、软件开发的周期等,使得利用计算机辅助数学研究成为常规的手段。例如在有限单群分类的研究中,对于散在单群的寻找,计算机就起到了关键的作用。

由于代数学侧重对数学对象结构上的刻画,以及代数学所具有的有限性、算法性和构造性等特点,因此在设计数学问题算法时,经常把问题转化为代数形式,计算机代数自然成为计算机数学的核心内容。例如在初等几何定理证明中,就是通过坐标的引进,将初等几何问题翻译成多项式方程组的形式,从而把定理的成立与否转换为对于多项式方程组零点结构的判定问题。又例如在微分方程解析解的计算中,也是将微分方程解的结构问题转化为微分代数或多项式代数的语言描述形式来设计相应的算法。

计算机代数的主要内容有归约计算、多项式计算和计算微分代数等几大部分,这里简单介绍域上多项式方程组(一元的和多元的)零点的计算和结构表示问题。

对于单变元多项式方程,四次(含四次)以下的可以根据求根公式写出符号解。四次以上的多项式没有一般的求根公式,目前计算机代数软件中实用的方法是将 $f(x)$ 通过因式分解表示为两个次数较低的多项式的乘积 $f(x) = g(x)h(x)$,再分别求出每一个因式的根;或者使用复合式分解,将 $f(x)$ 写成两个次数较低的多项式的复合 $f(x) = g(h(x))$,然后通过计算 $g(x)$ 和 $h(x)$ 的根得到 $f(x)$ 的根。通过辅以其他一些技巧,这个方法在很多情况下是有效和快速的。对于不存在公式解的方程,使用该方法自然是失效的。但在有些情况下,即便公式解存在,该方法也可能失败,因此该方法不是完备的。

多元多项式方程组的零点集一般是代数曲面,不是孤立的点,因此零点集的表示和计算都是复杂的问题,除了在一些特定情况下的算法外,在一般意义下给出零点计算和结构表示的方法首推吴方法和Groebner基方法。

吴方法: 设

$$F = \{f_1(x_1, x_2, \dots, x_n), f_2(x_1, x_2, \dots, x_n), \dots, f_m(x_1, x_2, \dots, x_n)\}$$

是域 A 上的多元多项式的集合。多项式 $f(x_1, x_2, \dots, x_k)$ 中所包含变量的最大下标 k 称为 $f(x_1, x_2, \dots, x_k)$ 的类,记做 $\text{Class}(f)$ 。从 F 中取出一个尽可能大的子集

$$G = \{g_1, g_2, \dots, g_p\}$$

使得 $\text{Class}(g_1) < \text{Class}(g_2) < \dots < \text{Class}(g_p)$,这个子集称为 F 的基列,用伪除法对 G 去逐个除 F 中的 f ,对于每一个 f ,得到表达式

$$Bf = h_1g_1 + h_2g_2 + \dots + h_pg_p + R$$

其中 B 是 g_i 的初式(首项系数) $\ln(g_i)$ 的幂的乘积, $B = \ln(g_1)^{p_1} \ln(g_2)^{p_2} \dots \ln(g_p)^{p_p}$, R 称为伪余式。如果 $R \neq 0$,将 R 添加到 F 中得到 $F' = F \cup \{R\}$,如果对于某个 g_k , $\text{Class}(R) = \text{Class}(g_k)$,则用 R 替换 g_k ,如果对所有的 $g_k, 1 \leq k \leq p$, $\text{Class}(R) \neq \text{Class}(g_k)$,则将 R 添加到 G 中去,得到的新的集合仍记做 G ,继续用新的 G 去除 F' 中的每一个多项式。该过程必终止(所有的 $R = 0$),终止时的多项式集合记做 \bar{F} ,基列记做 $\bar{G} = \{\bar{g}_1, \bar{g}_2, \dots, \bar{g}_q\}$,对于 F 中的任何多项式 f ,满足

$$\bar{B}f = h_1\bar{g}_1 + h_2\bar{g}_2 + \dots + h_q\bar{g}_q,$$

这样的基列称为 \bar{F} 的特征列, \bar{F} 与 F 生成的理想是相同的。记 $Z(F)$ 为多项式集 F 的零点集,则

$$Z(G) \setminus \left(\bigcup_{1 \leq i \leq q} Z(\ln(\bar{g}_i)) \right) \subseteq Z(\bar{F}) = Z(F) \subseteq Z(\bar{G})$$

特征列中的多项式可以按含变量的个数排成三角形形式

$$\bar{G} = \begin{Bmatrix} f_0(x_1, x_2, \dots, x_d) \\ f_1(x_1, x_2, \dots, x_d, x_{d+1}) \\ \vdots \\ f_{n-d}(x_1, x_2, \dots, x_d, \dots, x_n) \end{Bmatrix}$$

零点的结构和性质容易从 \bar{G} 导出,在零点个数有限的情况下($d=1$),零点还可以通过逐个方程的计算迭代求出。吴方法把复杂的方程组化成了简单明了的表示形式,并保留了方程组零点的许多重要性质,适应于以零点论式表述的数学问题,在几何问题计算、机器定理证明等领域能够借助吴方法设计快速和有效的算法。

Groebner基方法: 设

$$F = \{f_1(x_1, x_2, \dots, x_n), f_2(x_1, x_2, \dots, x_n), \dots, f_m(x_1, x_2, \dots, x_n)\}$$

是域 A 上的多项式的集合。对于 F 中多项式 $f(x_1, x_2, \dots, x_n)$,记

$$f(x_1, x_2, \dots, x_n) = \text{head}(f) + \text{tail}(f),$$

其中 $\text{head}(f)$ 是 f 的首项, $\text{tail}(f) = f - \text{head}(f)$ 。取 F

中任两个多项式 f, g , 构造

$$h = \frac{\text{lcm}(\text{head}(f), \text{head}(g))}{\text{head}(f)} f - \frac{\text{lcm}(\text{head}(f), \text{head}(g))}{\text{head}(g)} g$$

lcm 表示最小公倍式, 如果存在多项式 $p \in F$, 使得 $\text{head}(h)$ 是 $\text{head}(p)$ 的倍式, 即 $q\text{head}(p) = \text{head}(h)$ (q 是单项式), 则令 $h' = h - qp$ (这一步称为首项归约), 如此继续, 直到取得多项式 $h_{f,g}, h_{f,g} = 0$ 或者 $\text{head}(h_{f,g})$ 不是 F 中任何多项式首项的倍式。 $h_{f,g}$ 称为 f 和 g 的 s -多项式, 若 $h_{f,g} \neq 0$, 将 $h_{f,g}$ 添加到 F 中得到新的 F' , 继续对 F' 中的所有多项式对 (f, g) 计算 s -多项式 $h_{f,g}$, 并将其添加到 F' 中, 这个过程最终必停止, 停止时得到的多项式集合记做 G , 它是有限的, 称为 F 的 Groebner 基。记

$$\text{Head}(F) = (\{\text{head}(f) \mid f \in F\}),$$

即由 F 中的多项式的首项生成的理想 (称为 F 的首项理想), G 满足两条特征性质: ① $G) = (F)$; ② $\text{Head}(F) = \text{Head}(G)$ 。

一般地说, G 中包含的多项式是很多的, 但 F 与 G 的零点集是完全相同的, $Z(F) = Z(G)$, G 也可以表示为三角形形式

$$G = \left\{ \begin{array}{l} F_0(x_1, \dots, x_d) \\ F_1(x_1, \dots, x_d, x_{d+1}) \\ \vdots \\ F_{n-d}(x_1, \dots, x_d, \dots, x_n) \end{array} \right\}$$

但与吴方法不同的是, 这时 $F_i(x_1, \dots, x_d, \dots, x_{d+i})$ 是多项式集合。Groebner 基也给出了原方程组零点结构和性质的很好刻画。在零点个数有限的情况下 ($d=1$), 零点也可以通过逐步迭代的方法求出。由于 Groebner 基 G 保持了原多项式集合 F 的理想性质, 适应于以理想论式表述的数学问题, 例如代数学中关于理想的各种计算。

多元多项式方程组零点的计算是计算机数学中最基本的问题之一, 很多理论和实际问题都可以转化为多元多项式方程组零点计算与表示, 或者相关的理想结构的计算, 因此在很多应用场合都能够遇到多元多项式方程组的计算问题, 寻求更加快速高效的算法仍是当前研究的重点。

计算机数学的产生和发展, 使得数学演算和推理更加机械化和形象化, 催化了数学与其他科学和工程技术的结合, 传统数学的很多抽象研究成果, 借助计算机这种有力的工具, 找到了新的应用手段, 从而大大推动了计算机科学与数学自身的发展, 以及

在其他领域的更为广泛的和富有成效的应有。

参考文献

1. 林东尧, 李文林, 虞言林主编. 数学与数学机械化. 济南: 山东教育出版社, 2001
2. 吴文俊主编. 王者之路—机器证明及其应用. 长沙: 湖南科学技术出版社, 1999
3. Blum L, Cucker F, Shub M and Smale S. Complexity and real computation. Springer-Verlag New York Inc., 1998 (李廉)

jisuanji tuxing biaoizhun

计算机图形标准 (computer graphics standard) 计算机系统中应用程序与图形软件、图形软件与图形硬件之间的接口标准, 以及用于记录图形信息的数据文件格式的标准。

计算机图形技术在计算机辅助设计、办公自动化、地理信息系统、过程控制、仿真、图像处理、软件开发等领域中起着重要的作用, 它已经成为人机交互的一种主要手段。但由于图形输入、输出设备种类多, 工作原理和性能参数差别大, 因此图形应用程序的开发难度大、成本高, 比较难以具备良好的易移植性。

图形应用程序的易移植性意味着 4 个方面的要求: ①应用程序在不同系统之间的易移植性; ②应用程序与图形设备的无关性; ③图形数据的易移植性 (互操作性); ④程序员的易移植性, 即程序员不经重新培训就能为不同系统编制图形应用程序。为了实现这些要求, 系统中有 3 个接口必须实现标准化 (图 1)。

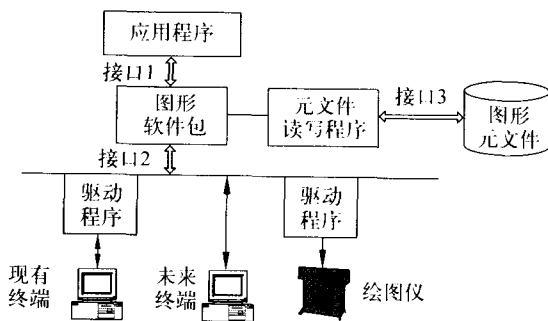


图 1 图形系统的 3 个接口

接口 1 是应用程序与图形软件的接口, 这是一个应用程序接口 (API)。图形软件是一组有关图形

处理的常用子程序的集合,它有效地隔离了应用程序与图形硬件设备之间的联系。因此,该接口的标准化就能实现应用程序在源程序级的易移植性。目前已经被批准为国际标准的有3个:第1个是图形核心系统 GKS(ISO/IEC 7942:1994);第2个是三维图形核心系统 GKS-3D(ISO 8805:1988);第3个是程序员的层次式交互图形标准 PHIGS(ISO/IEC 9592:1997)。3个标准中 GKS 是二维图形标准, GKS-3D 和 PHIGS 是三维图形标准,它们既有许多共同之处,又各有特色,互相补充。国际标准化组织还对上述3个标准分别制定了与 FORTRAN、Pascal、C 和 Ada 语言的语言联编标准。语言联编是图形功能调用的子程序名及参数的约定,它为应用程序开发人员提供了共同遵循的准则,目的同样是为了应用程序具有良好的易移植性。

由于国际标准制定的迟缓以及符合标准的图形软件速度慢、内存开销大,因此一些软件公司自行开发了图形软件,并在应用中得到了普遍认可,成为事实上的工业标准。例如, Silicon Graphics Inc (SGI) 公司的 OpenGL, Microsoft 公司的 Direct3D 等。

接口2是图形软件与图形硬件之间的接口,它也是一个程序接口,称为图形工作站接口。实现这个接口的标准化,就可以保证图形软件与图形设备之间的相互独立性。ISO 为这个接口制定的标准称为软件与图形设备对话的接口技术,即 CGI(ISO/IEC 9636:1991)。CGI 也可以作为不同图形设备之间的数据交换标准,它规定了 CGI 格式的数据流标准(ISO/IEC 9637:1994),符合标准的数据可以在具有 CGI 接口的设备(如图形显示器、绘图机等)上直接进行显示或打印。

Microsoft 公司在 Windows 操作系统中提供的图形设备接口 GDI 也是一种图形软件与图形硬件之间的接口,其功能是在显示器、打印机等图形设备上提供对文本、图形和图像输出的支持。GDI 屏蔽了不同输出设备各自的物理特性,使 Windows 应用程序能够在它所支持的任何图形输出设备上运行。

接口3是一种数据接口,它规定了记录图形信息的数据文件格式的标准。实现了图形文件格式的标准化之后,程序与程序或系统与系统相互交换图形数据就有了保证。ISO/IEC 制定的用于图形数据交换的国际标准是 CGM(ISO/IEC 8632:1999)。CGM 定义了二维图形(图片)信息的计算机可解释的表示,这种表示与任何特定设备或应用无关,其目的是便于图形数据的储存、检索以及在不同系统、不

同应用和不同设备之间的交换。近些年来, CGM Open 联盟与 W3C 合作开发了 CGM 的一种框架——WebCGM,使 CGM 图形元文件更适合在 Web 环境下应用。

实用中的图形数据交换格式还有多种。例如,在 CAD/CAM 软件中用于交换产品定义数据的美国国家标准 IGES(现已停止发展);法国和欧洲采用的工程数据交换标准 SET(现已被 STEP 取代);ISO 制定的产品数据表示与交换的国际标准 STEP(ISO 10303)中的几何与拓扑信息的表示(ISO 10303—42:2000);ISO 制定的在办公文档中嵌入矢量图形的标准(ISO 8613—8:1994) GGCA;W3C 制定的在 Internet 资源中使用 XML 语言描述二维矢量图形的标准 SVG。一些厂商也为自己的图形软件制定了专用的数据交换格式,如 Autodesk 公司的 DXF 格式、Microsoft 公司的 WMF 格式等,它们也得到了广泛的使用。

为了使各个图形标准规范化,明确各个图形标准之间的关系,ISO/IEC 还制定了一个 CGRM 标准(ISO/IEC 11072:1992),它并不针对图形系统中某一特定部分,而是整个图形系统的一个5层概念模型,目的是对复杂的系统从整体方面进行约束,使得各个单元技术的发展能够协调一致,可以说,它是一个图形标准的标准。

计算机图形标准已被工业部门和计算机用户所接受,它们已得到广泛的应用,已制定的图形标准将逐步完善,新的图形标准还将继续出现。

参考文献

ISO/IEC 11072:1992. Information Technology-Computer graphics-Computer Graphics Reference Model
(张福炎)

jisuanji tuxingxue

计算机图形学(computer graphics) 应用计算机生成、处理和显示图形的理论、方法和技术。它可以生成现实世界中已经存在的景物的图形,也可以生成虚构景物的图形。它所研究的主要内容是景物的几何建模方法,数字模型的绘制技术,图形输入和控制的人机交互界面,以及计算机动画。近年来,由于多媒体技术、计算机动画、科学计算可视化及纹理映射等的迅速发展,计算机图形学和图像处理的结合日益紧密,且相互渗透,进一步促进了这两个相关领域的发展。

早在20世纪50年代末至60年代初期,就出现

了图形显示器。1963年,美国麻省理工学院的 I. E. Sutherland 在他的博士论文中首次提出了交互式计算机图形学的概念。在他的 SKETCHPAD 系统中,不仅引入了分层存储符号和图素的数据结构,还可利用光笔实现选择、定位等交互功能。40年来,计算机图形学有了飞速的发展。在图形显示设备方面由60年代中期的随机扫描刷新式矢量显示器,70年代的存储管式显示器,到80年代后广泛采用的光栅扫描显示器,价格越来越低廉,显示功能越来越强大,性能越来越稳定。再加上个人计算机性能迅速提高和出现了低价位普及型的计算机辅助设计(CAD)及图形生成软件,使得计算机图形广泛融入到人们的科研、生产、工作、学习和日常生活中,成为加强信息传递和理解的有力工具。

计算机图形学目前的主要应用领域有:①计算机辅助设计和制造;②科学计算可视化;③**计算机仿真**;④计算机动画;⑤计算机艺术;⑥**计算机游戏**;⑦**办公自动化**及电子出版;⑧**地理信息系统**;⑨**影视特技与广告制作**等。

计算机图形学的主要内容有**造型技术**、图形绘制及**人机交互技术**3部分。要在计算机屏幕上生成三维景物的图像,首先必须在计算机中建立该景物的表示。这一技术称为造型技术。最常用的**几何造型技术**通过建立描述景物几何形状和拓扑结构的数据结构来表示所要显示的形体。几何形状的可控性及覆盖面是研究工作的重点。近年来,随着三维扫描技术的发展,一些外形复杂的景物大都改由大量三角形面片组成的多面体来表示。由于景物中三角形数量十分庞大,影响了画面实时绘制的速度和景物模型在网络中传输的效率。因此,基于三角形网格的景物多分辨率造型技术和几何数据压缩、传输技术成了几何造型近年来研究的热点。基于图像的造型技术则通过从一系列不同视点、视线方面拍摄的场景图像(含深度图像)来表示一个场景,进而生成场景在任意新视点处的画面。动画技术的发展,导致了**基于物理的造型技术**。自然界的许多景物很难或者根本无法用规则形体进行表达,如山脉、杂草、毛皮、云彩、波浪、浓雾等。为了表示这类自然景物,又出现了**分数维造型**,**基于文法的造型**等。

建立了景物的表示后,还需要根据观察者当前的视点位置和视线方向,生成该景物的真实感画面,这一过程称为**绘制**,它包括景物的取景变换、视域裁剪、消除当前画面中不可见的景物隐藏面,以及基于一定**光照模型**的景物表面**光亮度计算**等。光照模型

包括局部光照模型和整体光照模型,它们都是根据物理和光学的定律对景物表面光照明效果及表面对入射光的吸收、反射、折射情况的描述。常用的真实感图形绘制算法有扫描线算法、光线跟踪算法、光能辐射度方法等。一般说来,生成的景物画面逼真度越高,所需的计算时间就越长,因此,如何实现复杂景物真实感图形的实时动态显示就成为计算机图形学领域多年来追求的目标和研究的关注点。除了不断提高计算机硬件的运算速度及图形软件的效率以外,并行计算是一个重要手段。

三维景物造型、真实感景物画面生成都需要在一个操作方便、易学易用的用户界面下工作,包括:图元及造型方法的交互选择;形体、模型的交互操作;观察点、观察方向的交互设置;光照模型参数的交互选取;色彩的交互设定等。这就需要依据人机交互技术的理论和方法,设计出友善的用户界面以满足交互地生成图形的需要。近年来迅速发展的**虚拟现实技术**,是与造型技术、**真实感图形生成技术**及**人机交互技术**密切相关的。快捷方便的造型技术、实时动态的图形生成以及易学易用的交互技术是构造虚拟现实环境的基本和必要条件。而虚拟现实技术则是综合运用这3个方面的最新研究成果并加以发展的必然结果。

参考文献

1. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2000
2. 孙家广,杨长贵等. 计算机图形学(第三版). 北京:清华大学出版社,1999
3. D. F. Rogers 著. 计算机图形学的算法基础. 石教英,彭群生译. 北京:机械工业出版社,2001
(彭群生)

jisuanji wangluo

计算机网络 (computer network) 地理上分散的多台自主计算机互连的集合。自主计算机这一概念排除了网络系统中主从关系的可能性。计算机互连必须遵循约定的通信(网络)协议,由通信设备、通信链路及网络软件实现。计算机网络可实现信息交互、资源共享、协同工作及在线处理等功能。

发展简史

计算机的应用普及是从**微型计算机**的出现开始的。而新的发展阶段乃是网络,人们称网络就是计算机,深刻地反映了网络在计算机发展史中极为重要的作用和影响。

1969年美国国防部的国防高级研究计划署(DARPA)建立了全世界第一个分组交换网 ARPANET,即 Internet 的前身。这是一个只有四个结点的存储转发方式的分组交换广域网,是为了验证远程分组交换网的可行性而进行的一项试验工程。1972年在首届国际计算机通信会议(ICCC)上首次公开展示了 ARPANET 的远程分组交换技术。

分组交换不同于传统电信网中采用的电路交换,是存储转发交换方式中的一种交换方式,它将要传送的报文分割成许多具有统一格式的分组,并以此为传输的基本单元,一一进行存储转发的传输。和电路交换相比,分组交换具有线路利用率高、可进行数据速率的转换、不易引起堵塞以及具有优先权使用等优点。因此,它被广泛用于计算机网络。1976年国际电报电话咨询委员会(CCITT)制定了用于公用分组交换网的协议标准 X.25,进一步推动了公用分组交换网的发展。

在总结最初的建网实践基础上,DARPA 组织有关专家开发了 ARPANET 第三代网络协议——TCP/IP(参见 TCP/IP 协议集),并于 1983 年在 ARPANET 上正式启用。与此同时 UNIX BSD 版安装了 TCP/IP 协议软件。TCP/IP 协议的广泛采用是 Internet 迅速发展的重要原因之一。1974 年 IBM 公司首先公布了系统网络体系结构(SNA)作为 IBM 计算机的连网标准。之后,各大计算机厂商都相继开发了自己的网络体系结构,如 DEC 公司的数字网络体系结构(DNA)等。为了解决不同厂商的计算机网之间不能互连的问题,国际标准化组织(OSI)于 1978 年提出了开放系统互连基准(参考)模型(OSI/RM),即 OSI 网络体系结构,以推动网络标准化工作。

1976 年美国 Xerox 公司开发了基于带碰撞检测的载波监听多路访问(CSMA/CD)原理,用同轴电缆连接多台计算机的局域网,取名为以太网。由于以太网安装使用方便,性能较好,成为最广泛使用的一种局域网。随着个人计算机(PC)的广泛使用,局域网的研究、开发和应用有了很大的发展。

Internet 是全球最大的、开放的、由众多网络互连而成的计算机网络。Internet 的发展已经历了三个阶段,逐渐走向成熟。从 1969 年 Internet 的前身 ARPANET 的诞生到 1983 年是 Internet 的研究试验阶段,它主要是作为对网络技术进行研究和试验用的网络。从 1983 年到 1994 年是 Internet 的实用阶段,作为用于教学、科研和通信的学术网络在美国和

一部分发达国家的大学和研究部门中得到广泛应用。从 1994 年以后,Internet 开始进入商业化阶段,除了原有的学术网络应用外,政府部门、商业企业以及个人广泛使用 Internet,且全世界绝大部分国家都纷纷接入 Internet,这种迅猛发展的进程反映了 Internet 正日益成熟。当前 Internet 技术和应用的超常规速度发展,对信息技术的发展、信息市场的开拓以及社会信息化起着十分重要的作用。由于 Internet 的社会需求飞速增长,它正面临着多种挑战,包括网络的频宽和可扩展性、网络的安全性、网络的服务质量、多种新的网络应用需求以及引发的商业、文化和社会问题。

网络分类

网络分类方式繁多,一般有以下几种:

- (1) 按地域范围可分为局域网、城域网和广域网 3 类;
- (2) 按拓扑结构可分为总线、星状、环状、网状等网络;
- (3) 按交换方式可分为电路交换网、分组交换网、帧中继交换网、信元交换网等;
- (4) 按网络协议可分为采用 TCP/IP、SNA、SPX/IPX、AppleTALK 等协议的网路;
- (5) 按应用规模可分为内联网、外联网等。

基本内容

随着计算机网络应用的日益广泛,网络规模的日益扩大,都对网络硬件和软件提出了新的要求,因此,计算机网络技术也得到了高速的发展。网络主要技术包括数据通信、体系结构与协议、互连、管理与网络安全等。

数据通信

自 20 世纪 80 年代以来计算机科学和数据通信的融合,大大改变了这些领域的技术和产品,推动了计算机-通信产业的发展。

正在融合的计算机-通信产业从元件装配发展到系统集成,而集成的系统又发展成能传输和处理多种形式的数据和信息的系统。技术和技术标准组织也正在走向一个能够集成所有通信的单一公共系统,使得全球所有的数据和信息源能够被方便地、用相同的方式访问和获取。

网络体系结构与协议

网络体系结构是计算机之间相互通信的层次以及各层次中的协议和层次之间接口的集合。网络协议是计算机网络和分布系统中相互通信的对等实体间交换信息所必须遵守的规则的组合。采用 TCP/

IP 协议集的网络体系结构是计算机网络体系结构的主流,并仍在改进和发展中。**网络协议工程**是一门研究如何设计和构造协议规范,以及如何把所设计和构造的协议规范快速、准确、低成本地转化为可执行代码的工程。

局域网

局域网是将小区域内的各种通信设备互连在一起的通信网络。传统局域网的典型特性是数据速率在 $0.1 \sim 100 \text{ Mb/s}$, 距离在 $0.1 \sim 25 \text{ km}$, 误码率在 $10^{-8} \sim 10^{-11}$ 。决定局域网特性的主要技术有三个:用以传输数据的**传输介质**;用以连接各种设备的**局域网拓扑结构**;用以共享资源的**局域网介质访问控制方法**。流行的传统局域网是以太网和**令牌环网**。近年来,对高速局域网的需求日益增加,相继出现了一些新的局域网技术。 100 Mb/s 速率的局域网有**光纤分布数据接口(FDDI)**和**快速以太网**。**千兆位以太网**是在传统以太网和快速以太网基础上发展起来的一种**高速局域网**,它仍然使用 CSMA/CD 介质访问控制方法,但将数据速率提高到了 1000 Mb/s 。交换式局域网采用不同于共享介质的访问控制方法,以提高局域网速率。此外无线局域网也是很有应用前景的一种局域网。

广域网

广域网是地理范围从数十千米到数千千米,可以连接若干个城市、地区,甚至跨越国界,遍及全球的一种计算机网络。基于 X.25 协议的公用分组交换网是早期最流行的广域网,它是基于分组交换方式的存储转发式网络,可提供中、低速数据通信业务。近年来,高速、宽带广域网的需求日益增加,光通信技术又有了很快发展,推动了快速分组交换技术的发展,其特点是简化通信协议和发展高速交换设备。目前广泛采用的技术有**帧中继技术**和**异步传送模式(ATM)**。帧中继技术是在数据链路层实现网络资源统计复用的一种快速分组交换技术。异步传送模式是以分组交换为基础并融合电路交换高速化,并以信元为单位进行标记复用的一种高速传送与交换技术。此外,**移动通信网**、**卫星通信网**也是很有应用前景的广域网。

网络互连

网络互连将多个网络互相连接以实现在更大范围内的信息交换、资源共享和协同工作。Internet 就是由成千上万个不同的网络互连而构成的**网际网**,或称**互连网**。**网络互连协议**是计算机网络间互相连接进行通信时有关数据格式及交互过程必须遵循的

约定。IP 是一种最著名的网络互连协议,由 Internet 广泛采用(参见**网际协议**)。**路由选择**是在网络环境中寻找一条到达目标计算机通路的过程。实现路由选择的算法称**路由选择算法**,并有相应的**路由选择协议**。常用的**网络互连设备**有**中继器**、**网桥**、**路由器**、**网关**。

网络管理

网络管理功能包括对网络的配置、故障、性能、安全、计费等进行管理的功能。**OSI 管理体系结构**是基于开放系统互连环境对资源进行管理的一种体系结构,它提供了在开放系统互连环境中控制、协调和监视各种资源的手段。**简单网络管理协议(SNMP)**是基于 TCP/IP 协议的一种功能比较简单的网络管理协议,被广泛用于 Internet。**公共管理信息协议(CMIP)**是由国际标准化组织(ISO)为开放系统互连(OSI)制定的网络管理协议标准。

网络安全

网络安全是在分布式计算环境中,对信息的传输、存储、访问提供安全保护,以防止信息被窃取、篡改和非法操作。网络安全的三个基本要素是**保密性**、**完整性和可用性**服务。在分布网络环境下还应提供**鉴别**、**访问控制**和**抗否认**等服务。完整的信息安全保障体系应包括**保护**、**检测**、**响应**、**恢复**四个方面。常用的安全防范技术包括**身份鉴别**、**访问控制**、**完整性控制**、**密码技术**、**防火墙系统**、**计算机病毒保护**、**审计和恢复**、**操作系统安全**、**数据库系统安全**等。

发展趋势

计算机网络的发展趋势可概括为:一个目标、两个支撑、三个融合、四个热点。

目标

面向 21 世纪计算机网络发展的总体目标就是要在各个国家,进而在全球建立完善的信息基础设施。信息基础设施将改变人们的生活、学习、工作、人际交往的方式,减轻人们的工作负担,提高人民的生活质量,推动社会的进步。

1993 年美国制定了**信息高速公路**(即国家信息基础设施(NII))发展计划后,各国政府都相继规划和实施 NII 计划。NII 的建设目标是在全国范围内建立为民众普遍服务的信息基础设施。NII 的基本组成包括**通信网**、**计算机**、**信息内容**和各种年龄、背景的人。NII 的层次结构可分为**传输层次**、**网络层次**、**终端系统**和**信息服务**这四个层次。NII 不可能在一个国家孤立地实现,因此,1994 年在西方七国部长会议上又提出了实施全球信息基础设施

GII 的若干原则。1998 年美国又提出了实施数字地球的计划。

NI 的建成是指一个国家的信息网络,能使任何人在任何地点、任何时间,可将文本、声音、图像、电视信息等各种媒体信息传递给在任何地点的任何人。

支撑

在实施面向 21 世纪计算机网络发展的总体目标时,有两个重要的支撑技术,即微电子技术和光技术。

微电子技术的发展是信息产业发展的基础,也是驱动信息革命的基础。其发展速度是惊人的,用摩尔定理来预测,微电子芯片的元件数量每 18 个月提高一倍。

这一发展趋势到 2010 年将趋于成熟,那时每个芯片上最多可包含 10^{10} 个元件,理论上的物理极限是每个芯片可包含 10^{11} 个元件。

对于典型的传统逻辑电路,每个芯片可包含的元件数少于 10^8 到 10^9 。每个芯片的实际元件数可能因经济因素的限制而低于物理上的极限值。

自 1980 年以来,微处理器的速度一直以每 5 年 10 倍的速度增长。PC 的处理能力在 2000 年已达 1 000 MIPS,预测在 2011 年可达 10 万 MIPS。

在预测网络性能增长时,有个 Metcalfe 定理,该定理预测网络性能的增长是连到网上 PC 能力的平方。这表示网络频宽的增长率是每年 3 倍。不久的将来,称为 Petabit,即每秒 10^{15} 位的网络频宽将出现。

新的微电子技术工艺正在开发的一种称为 Cu 的芯片,具有低阻抗、低电压、高计算能力的特点。IBM 研制的第一块 Cu 芯片可运行在 400 ~ 500 MHz,包含有 150 ~ 200 百万个晶体管。另一种用紫外平面印刷技术的 EUV 工艺是 $0.1 \mu\text{m}$ 的新一代的芯片制造技术。Intel、AMD、Motorola 公司都投入了巨资进行研究。Intel 期望在 2011 年能生产每个芯片包含 10 亿个晶体管的产品。

驱动信息革命的另一个支撑技术是光技术。评价光纤传输发展的标准是传输的比特率和信号在需要再生前可传输的距离的乘积。在过去 10 年间,该性能每年增长一倍,这种速度可望再持续 10 到 15 年。

第一代光纤传输使用 $0.8 \mu\text{m}$ 波长的激光器,传输速率可达 280 Mb/s。第二代使用 $1.3 \mu\text{m}$ 波长的激光器和单模光纤,传输速率可达 560 Mb/s。第三

代使用单频 $1.5 \mu\text{m}$ 波长的激光器和单模光纤。目前使用的第四代采用光放大器,数据传输速率可达 10 Gb/s 到 20 Gb/s。

光放大器的引入给光纤传输带来了突破性的进展。采用波分复用技术对于传输容量的提高有很大影响。一个 40 Gb/s 的系统能在同一光纤中传送 16 种波长,每一波长速率为 2.5 Gb/s。因为允许所有波长同时放大,光放大器能提供很大的容量。在单一光纤上传输 100 Gb/s 含 40 种波长的商用系统已在 2000 年实现,该系统可同时传送 100 万个语音和 1 500 个电视频道。

融合

支持全球建立完善的信息基础设施的最重要技术是计算机、通信、信息内容这三个方面的融合。计算机包括计算机硬件、计算机软件以及相应的服务;通信包括电话、电视电缆、卫星以及无线通信等;信息内容包括教育、娱乐、出版、信息提供者等。

信息时代的经济发展要求计算机、通信和信息内容三种关键经济成分的融合。在企业组织结构上,负责这三方面的组织也应融合。在技术上,应改变过去三种技术分离各自发展的状况,使它们成为集成的技术。

电信网、电视网、计算机网三种网络的合一是当前网络发展的趋势。20 世纪 70 年代末,综合业务数字网(ISDN)概念的提出,表明人们期望有一个能提供数据、声音等多种媒体信息的综合业务的网络。但是多年的实践表明 ISDN 的发展没有获得预期的结果。其原因固然有技术不成熟的因素,更重要的是缺乏市场的驱动。

近年来在市场的驱动下,数据通信得到了迅速的发展。在美国,数据通信的通信流量已经超过语音的通信流量,电信部门纷纷以数据业务为中心来重组网络,使之适应以数据业务为中心的多业务服务的需要。电信、有线电视、计算机和信息行业的许多兼并现象使这些行业间的界限越来越模糊。行业融合是市场驱动的结果,反过来它又进一步推动市场的发展。

计算机、通信、信息内容的融合,电信网、电视网、计算机网的合一,其最重要的技术基础是数字化。人类正快速进入数字世界,所有通信方式,包括语音、图像和数据都能够而且都将被数字化,这样信息之间的转换、重组、存储、处理、传输都变得十分简易,为各种信息交互的模式提供了基础。

计算机网从一开始就是基于数字信号技术的。

电视将变成存于多媒体数据库的数字文件。根据用户的要求调用这些数字文件,即可实现点播电视功能。电话从最初采用数字交换设备发展到完全由数字信号传输。数字化不仅改善了通信的质量,而且使各种媒体交互功能成为可能。

在实现三网合一中,一个十分值得重视的问题是网络体系结构。从发展的趋势看 Internet 体系结构是十分有前途的。如前所述数据通信的业务将大大超过话音通信的业务,而在数据通信中,Internet 已成为主流。“IP over every thing”和“every thing on IP”。这两句话很好地概括了 IP(参见网际协议)的重要特征。以 IP 为核心的网络体系结构,向下可兼容各种不同的通信机制,如异步传送模式(ATM)、帧中继(FR)、同步数字复用体系(SDH)、波分多路复用(WDM)等技术(参见帧中继交换、时分多路复用);向上可实现各种不同的业务,如多媒体业务、视频和音频组播业务、有线电视、直播卫星、无线电和广播电视等业务。

在三网合一的过程中,采用 Internet 体系结构和 IP 协议时应提高网络的可靠性。因为目前数据网的可靠性达 99.9%,而电话网的可靠性可达 99.999%。当前研究工作者和厂商正在致力于提高其可靠性和服务质量。

热点

多媒体 随着数字化技术的成熟,数据、文本、声音、图像这些媒体都能数字化,导致多媒体技术的出现。人们在事务处理和日常生活中,本来是将各种媒体的信息集成在一起的,所以多媒体技术更加接近于人的生活。

多媒体包括静态的和基于时间的媒体,前者是数据、文本和静态图像,后者是音频和视频信号。实时动画类似于视频信号,而实际上是一系列按序存储且能快速显示的帧,其内容由软件控制并能很快再现。

另一种基于时间的媒体是虚拟现实。它是实时动画的扩展,人进入到里面好似进入了一个真实的世界,而实际上进入的是一个智能联网的空间。

多媒体的应用有视频点播、交互视频,包括视频的协同工作、文件共享、白板、远程医疗和远程教学。多媒体应用将极大地影响人们的生活。

所有多媒体应用的共性是需要很宽的传输频带和处理能力,越来越复杂的应用需要越来越宽的频带和越来越高的处理能力。多媒体应用是促进技术和行业融合的强大市场驱动力。

宽带网 从传统电话网到光载波信号线路 48 级(OC 48)的光缆通信,相当于从 1 米宽的乡村小道发展到 32 千米宽的超高速公路。而采用先进的压缩技术,又可以在同样带宽的信道上传输更多的信息。

要建立真正的宽频多媒体网络,达到信息高速公路的目标,其费用是十分昂贵的。一方面要有高速的传输载体,即信息高速公路的物理结构,包括网络、软件、交换设备。另一方面还要有通过载体传输的信息内容。

信息高速公路的载体有两个技术特征:一个是在任何时间、任何地点要能提供全彩色、全动态的视频信号,一般称为宽频容量;另一个是要提供全交互的、双向的信息流通信。传统的电话技术可在全世界范围实现双向通信,但接入最终用户的容量有限。传统的电缆网容量大,通常一条电缆可传输 12 个视频通道,但电缆网是单向传输,没有交互通信能力。目前已有一些技术,可提高电话网接入系统的带宽和使电缆网实现双向通信。

除了电话、电缆、光缆以外,还有很多通信技术也在发展成为信息高速公路的载体,如直接广播卫星、蜂窝电话、低轨道交互卫星等。但是,基于大容量光缆、SDH 以及 WDM 的宽带技术是建立信息高速公路的主流技术。更高带宽的 IP 交换和 ATM 交换是解决目前路由器瓶颈的主要技术。

以太网的历史已有 20 多年,由于其固有的优点,在局域网中使用最广泛,其性能也不断改善。速率已从最初的 10 Mb/s 发展到 100 Mb/s,进而到 1 000 Mb/s。IP 在以太网上运行具有很高的效率,其应用范围已从局域网扩展到城域网,1 000 Mb/s 以太网无转发的传输距离已经达到数十千米,甚至超过 100 千米。

长期以来,接入是网络的一个瓶颈问题。将宽带业务带进每一个家庭,需要解决宽带接入方法(参见宽带接入技术),即所谓的最后一英里问题。不对称数字用户专用线(ADSL)(参阅数字用户专用线)使用标准的一对电话线,它的特点是接收信息的速率大大高于发送信息的速率。采用 HFC 结构和电缆调制解调器或机顶盒,也是一种很有发展前途的技术。从长远角度来考虑,应该采用无源光网 PON,它允许在户外的设备中对波长进行简单的分割,从而为每个家庭提供全部的波长,通过它可实现真正的端到端的多媒体通信。

移动通信 个人通信系统(PCS)可以使用无线

技术,在任何地方以各种速率与网络保持联络。用户利用 PCS 进行个人通信,在任何地方都能接收到发给自己的呼叫。这些 PCS 系统可以支持语音(言语)、数据和报文等各种业务。PCS 网络和无线技术将提高人们的移动通信水平,成为未来信息高速公路的重要组成部分。

随着增加频谱,采用数字调制,改进编码技术和建立微小区和宏小区,在未来十年里,无线系统的容量将增加 1 000 倍以上。系统的容量通过使用动态信道分配技术还能得到进一步的增长。利用自适应无线技术,将包含电子信息的无线电波束信号发送到接收方,并将清除其他的干扰波束,从而降低了干扰,提高了系统的容量和质量。

第一代无线业务有两大类:一类是蜂窝-个人通信服务(PCS)广域网,提供话音业务,工作在窄带,服务区被分成宏小区;第二类是无线局域网,工作于更高的带宽,提供本地的数据业务。

新一代的无线业务将包括新的移动通信系统和以宽带信道速率(64 Kb/s ~ 2 Mb/s)在微小区之间进行的固定无线接入业务。使用可控无线固定天线业务可以在同样的频带内提供更高的带宽。

信息安全 当前,网络与信息的安全正在受到严重的威胁,一方面是由于 Internet 的开放性以及安全性不足,另一方面是由于众多的攻击手段,诸如病毒、陷门、隐通道、拒绝服务、侦听、欺骗、口令攻击、路由攻击、中继攻击、会话窃取攻击等难以防护。以破坏系统为目标的系统犯罪和以窃取、篡改信息、传播非法信息为目标的信息犯罪都会对国家的政治、军事、经济、文化造成严重的损害。为了保证信息系统的安全,需要完整的安全保障体系,该体系应具有保护功能、检测手段、对攻击的反应以及事故恢复能力。

参考文献

1. 胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002
2. Andrew S Tanenbaum. Computer Networks (third edition). Prentice Hall, 1996 (胡道元)

jisuanji weihu

计算机维护 (computer maintenance) 为保证计算机系统正常、高效运行而进行的各种调整、检测、监控、维修及保养等工作。计算机维护是计算机投入运行后必须进行的日常工作,它对提高计算机系统的运行效率,保证数据(信息)的安全与完

整,延长计算机设备的使用寿命都有十分重要的作用。

计算机系统的运行维护 计算机系统安装调试完成后,一旦投入运行,系统维护即成为不可缺少的日常工作。系统的运行维护的目的是为了使系统更有效地运行并保证系统中信息的安全与完整。

资源利用监控 系统维护人员必须经常监视计算机资源(如中央处理器、内存存储器、输入输出通道等)的运用情况并进行必要的均衡与调整,以使有限的资源得到最大限度的利用。某些情况下仍需要人工进行干预,以均衡不同作业对资源的利用,消除影响系统高效运行的各种“瓶颈”。由于多个作业争夺资源出现的“死锁”,或某些作业由于程序缺陷而出现的无休止循环都会造成资源的浪费,均需要人工干预予以排除。

作业管理 对多用户多任务的应用系统,作业管理的任务包括根据作业的需要为其分配资源并适时予以回收;根据待处理作业的不同类型,对资源的不同需求以及当前运行情况进行作业的调配;进行分时时间片的设置以及定时启动或中止某些作业的运行;给定与调整不同类型作业的优先级,如白天给予交互式作业高的优先级,晚间与假日则将处理机资源主要分配给大型批处理作业等。

信息转储 系统运行过程中不断产生大量信息,包括系统本身的运行状态信息及在系统中运行的各种作业信息。为了保证信息的安全性与完整性,也为了充分发挥高速存储器(内存存储器及磁盘存储器)资源的作用,各类应用系统都必须根据具体情况,从内存存储器或磁盘存储器向磁带或软磁盘、光盘等介质转储系统运行的状态信息及作业的有关信息。对大型连续运行的应用系统,转储的频度至少每天一次。即使是微型计算机系统,也需经常进行信息转储,例如在磁盘空间将用满或一些不常用的文件需要保存的情况,就应将盘空间的信息转储到软磁盘。转储的介质应存放于安全处所,对特别重要的转储信息,应有两份拷贝存放于不同地点,以防止因系统故障及其他突发事件导致大量信息丢失而造成严重的后果。

用户服务 对于大型分时系统,用户服务是一项不可缺少的维护工作。如用户的登录与注销;接受用户的咨询,使用户了解人机之间的界面;根据用户的请求以相应介质输入、转储或输出信息,并为用户保存信息存储介质;通过系统控制台与用户通信,在系统发生突然情况时向用户通告,以便采取必要

措施妥善处理作业的异常中止等等。

计算机硬件的预防性维护 计算机硬件是由大量光、机、电元器件集合组成的装置,元器件数量巨大且通常处于连续运行的状态,不可避免地存在设备老化及寿命问题。为了尽可能降低运行中出现设备故障的几率,提高系统的可靠性,应采取以预防为主维护措施,对偏离正常状态或接近使用期限的元器件、零部件重新调整其工作状态或予以更换,以保证系统连续正常地运行。

系统初始启动的例行检测 一个准备投入运行的系统,在系统加电后首先运行基本测试程序,检查系统各个部分是否正常工作,如未发现异常即可进入运行状态,否则即停机检修后再投入运行。较先进的系统在运行中间,其诊断部件会自动对系统中的关键部件进行巡回检测,并对检测结果予以记录或显示,系统维护人员通过监视检测情况可及时了解运行状态,必要时采取措施防止错误发生或扩大。

定期进行系统的全面检测 对较复杂的计算机系统,应定期进行系统的综合性诊断检测程序,并分析近期的运行记录,判断系统是否存在隐患,是否发生过偶发性故障,某些部件的组成部分是否已被自动切换。目前较先进的计算机系统均采用容错技术以提高可靠性,如通过复执手段纠正逻辑操作的偶发性错误;通过校验手段纠正存储器的单个位错误;对某些功能相同的寄存器及存储器阵列设置备份,必要时进行自动切换等。这些系统自动进行的维护措施会在系统的运行记录中有所反映。维护人员根据综合测试及分析运行记录的结果采取更换设备或重新配置等措施,使系统保持在良好的工作状态。一般大中型系统,每周应进行1次~2次全面检测,小型微型系统可适当延长全面检测的时间间隔。

定期进行机电设备的维护保养 对易损易污的机电运转部件(如磁带机、打印机等)、通风过滤装置进行清洁、润滑或更换零配件;调整敏感元器件的阈值电压及基准电平;调整精密装置如软磁盘机的读写磁头运动机构相对于盘片零道的位置;调整真空部位(如磁盘机、磁带机的气动部位)的气压等。上述设备的维护保养对种类繁多结构复杂的外围设备尤其重要。外围设备是计算机系统发生故障最多的部分,做好设备的维护保养可大大降低整个系统的故障率。一般应一个月进行一次局部的机电设备维护保养,一季度至半年进行一次全面的机电设备零配件更新与维修。易损易污部件的清洁工作(如磁带机读写磁头沾染的磁粉擦抹,打印机走纸机构的

纸屑清除等)则应每天进行。

计算机发生故障后的临时性维修 如计算机发生故障导致系统不能运行则应停机进行临时性维修。首先要区分是软件故障还是硬设备故障。软件故障可能是因为系统软件的某个环节在特定组合条件下不能正常运行引起的,也可能是多种作业在运行中因争夺资源而出现死锁等原因造成的。这类故障一般可采用重新启动系统或其他人工干预手段予以排除。如果是设备性能变坏引起的硬件故障,则应使系统从运行状态转为故障诊断状态,运行诊断测试程序检测全机各部件,特别是中央处理机和磁盘存储器两种部件(输入输出部件一般不至于影响整个系统的正常运行),尽快故障定位,然后针对故障部位进行维修。

中央处理机的临时性维修 中央处理机的故障原因主要是集成电路失效。当前的计算机系统均配备较完善的诊断测试手段,提供详细的故障维修指南,对大部分电路的故障可以实现准确定位。但由于集成电路组装密度很高,一个集成电路芯片包含的逻辑单元或存储单元数以百万计,诊断测试程序检测出的故障通常定位在一个电路模块或一个乃至几个电路卡,维护人员根据测试结果能在现场进行的维修工作就是更换电路卡。如现场没有相应的备份件,可以采取降级运行(如多处理机系统可切除有故障的处理机,存储器可切除部分有故障的单元等)的手段使系统保持连续运行,若没有补救手段则需停机检修。

外围设备的临时性维修 对外围设备的故障检测应采用脱机检测与联机检测两种方式。脱机检测是指外围设备在逻辑上与中央处理机脱离联系(必要时把物理连接也脱离)的情况下对不同外围设备运行特定的测试程序,进行不含接口部分的功能测试,借助设备的面板或专用测试器具显示的信息并参阅维修手册判断故障所在部位。外围设备的故障有一类是集成电路失效,可通过更换电路卡排除。另一类是各种外设的特殊故障,常见的如磁盘盘面损伤,读写磁头位置偏离或其运载机构不能正常运动,打印机的打印部位损坏或打印纸传递机构故障等,需根据具体情况进行维修。如脱机测试正常而联机却不能正常运行,则应进行针对该设备的联机测试,运行相应的测试程序,测试该设备与中央处理机的接口部位并检验两者之间的协调关系。必要时还可进行模拟环路测试,即将外围设备至主机之间的输入输出连线构成回路,以确认故障所在部位是否在接

口电路。

电源部件的维修 计算机硬件中的各个部分均有专用的电源部件,电源部件中有一部分是大功率的分立器件,故障率较高,是硬件中常见的故障部位。系统维护时除了定期对这部分元器件进行检修更换外,在检测中央处理机及各种外围设备时,如发现工作异常,应充分注意到电源部件是可能发生故障的主要部位。

随着计算机技术的发展及各种应用对计算机系统可靠性、可用性不断提高的要求,今后的计算机系统将更广泛采用冗余技术,如动态重配置、磁盘镜像等技术,尽可能避免由于硬件故障而引起的系统运行中断。这样既提高了系统的可靠性和可用性,也降低了临时性维修的难度。(苏振泽)

jisuanji xitong kekaoxing

计算机系统可靠性 (reliability of computer system)

在规定的条件下和规定的时间间隔内,计算机系统能正确运行的概率。“规定的条件”包括环境、使用、维修等条件和操作技术。“规定的时间”是指可靠性是对一定的时间间隔而言,人们常常要求能在规定的时间内具有一定的可靠性。“正确运行”是指系统能完成规定的各项技术性能;运行的程序不被破坏或停止;程序按规定的要求运行;运行结果正确;执行时间不超过一定限度等。

计算机系统可靠性通常用平均故障间隔时间 (MTBF),即系统能正确运行时间的平均值来表征。

系统可靠性一直是计算机的一项重要指标。随着计算机应用范围的扩大,要求计算机系统不但有良好的技术性能,同时还必须有较高的可靠性。提高系统可靠性一般有两类技术方法,即避错法和容错法。

硬件避错技术的作用是减少失效的可能性。主要有:①对元器件进行老化筛选。②使用可靠的连接组装技术,严格工艺生产过程中的质量控制。③在设计时对元器件的额定参数留有足够余量,即电应力、热应力的实际使用值明显地低于额定值,通常只有额定值的几分之一。④由于元器件参数的离散性和负载变化产生的参数漂移,在基本节拍内逻辑链级数延时的设计留有足够余地。⑤降低系统内部的电磁干扰。如印制电路板中,限制平行走线、重叠走线、分叉走线的长度和数量;限制接插件上同时同相动作信号线的数量;采取地线和地平面、电源线 and 电源平面等隔离技术以及电源的高、低频滤波等。

⑥高速信号线采用传输线及匹配技术。⑦屏蔽外界电磁干扰。⑧做好热设计,必要时采取冷却措施(风冷、水冷),降低机柜内的温度。⑨采取防震、防冲击、防潮、防盐雾等机械结构措施。通过上述避错技术可使系统能承受一定的工作条件的变化,如电源拉偏、频率拉偏、温度变化、电磁干扰、机械震动等。

提高可靠性的另一种方法是容错法,主要采用硬件冗余、软件冗余、信息冗余和时间冗余等技术,屏蔽故障的影响。使系统出现故障时,仍能保持正常的功能。

硬件容错技术包含故障诊断、故障屏蔽和动态冗余技术。故障诊断是检测系统故障的发生及其确切位置。它是实现故障屏蔽和动态冗余的先决条件。故障屏蔽是一种静态硬件冗余技术。它通过冗余资源来隔离或校正故障的影响。如双工冗余、模3表决冗余、海明码纠单错等。屏蔽技术只是容忍一定程度的故障,当冗余资源耗尽后,再发生故障,系统就会产生错误的结果。

动态冗余是综合性的容错技术,已广泛用于容错计算机系统中。其基本思想是:在发生故障时,通过系统内部的自动重组来切除故障部件,并以备用部件替换故障部件。实现动态冗余,要求系统具有模块化结构和故障检测、定位功能。用检测到的故障信号激活故障处理程序,自动将故障部件处理的任务转移到完好的部件或备用部件,并启动系统运行或性能降级运行。故障部件可以联机维修或脱机维修(使用活线拔插技术),修复后再加入系统,不中断系统运行。动态冗余有电路级、分系统级和系统级冗余。电路级冗余指在系统电路的关键部分设置备份电路,在发生故障后启用。分系统级冗余是指分系统装置多重化,如采用冗余多处理器系统,可互为备份或采取服从多数裁决的策略。其他如输入输出部件、存储部件、系统互联网等都可设置冗余。系统级冗余则有双工系统、均分负载系统和双机系统等。

时间冗余又称“复执”,通过重复执行发生故障的操作,使由于某种不明原因而引起的瞬时偶发性故障不再出现。复执有硬件复执和软件复执两种。硬件复执由增设的控制电路,在出现故障后,延迟若干时间,重复执行发生故障的操作,如此时偶发性故障已消失,则复执结果可获成功。延迟的时间长短和1次故障需要复执的次数可根据经验设定。软件复执可在单条指令、1段程序或整个作业3个层

次上进行,复执方案根据瞬时偶发性故障的多少、程序的检查点设置、复执所付出的代价以及复执的成功率等来决定。

为了保证系统的可靠性,在系统的设计阶段就应将可靠性作为一项重要的设计内容,除了采用上述的避错和容错技术方法外,还应对系统进行可靠性分配与可靠性预测。

可靠性分配是根据用户对计算机系统可靠性指标的要求,对各分系统、设备等各部分提出相应的可靠性指标。并逐级分配到集成电路、元器件、接插件、印制电路板、生产工艺质量等项目上。在实际设计中,一般要提出多个方案进行比较、调整,以求得合理的结果。

可靠性预测是对系统的可靠性进行估算。简单的计算方法是先求出分系统的失效率 λ_i :

$$\lambda_i = m_1 \lambda_{i1} + m_2 \lambda_{i2} + \cdots + m_K \lambda_{iK}$$

式中, K 为组成第 i 个分系统的成分种类,如集成电路、元器件、接插件、印制板金属化孔、焊点等; m 为各种成分的数量, $m_K \lambda_{iK}$ 为第 K 种成分(例如焊点)的失效率。

再求出系统的失效率 λ :

$$\lambda = n_1 \lambda_1 + n_2 \lambda_2 + \cdots + n_L \lambda_L$$

式中, L 为分系统(部件、设备)的种类; n 为分系统的数量; $n_L \lambda_L$ 为第 L 种分系统的失效率,则系统的平均故障间隔时间 MTBF:

$$MTBF = \frac{1}{\lambda}$$

MTBF 的单位为 h(小时)。在实际使用时,一般还要考虑:①系统所处的环境;②反映设计成熟程度的修正因子;③反映分系统冗余程度和容错能力的加权因子等。预测的失效率与实际的故障率有一些差异是正常的,设计阶段的可靠性预测只是一种估算。

计算机系统可靠性技术是计算机科学工程的一个重要领域。由于超大规模集成电路的发展,出现了各种类型高性能计算机。计算机互连和数据传送的高可靠性技术和计算机系统的其他高可靠性措施都将有很大的发展。系统的正确性证明、系统故障的自诊断、自修复技术、软件容错技术和系统的高保密性、新的防病毒措施等,将取得进一步发展。新的“永不停机”的计算机系统即将出现。

参考文献

1. 傅佩琛,赵霖,张军英. 计算机系统硬软件可靠性理论及其应用. 北京:国防工业出版社,1990

2. R. 朗博顿著. 计算机系统可靠性. 张复,吴仲贤译,游鄂毓校. 北京:国防工业出版社,1988

(陈玉霜 应秋丽)

jisuanji xitong xingneng moni

计算机系统性能模拟 (performance simulation of computer systems)

对计算机系统,采用以软件模拟其操作的方法来评价其性能的一种技术。它通过模拟程序的运行动态地表达计算机系统的状态,并进行统计,得出系统的性能指标。计算机系统性能模拟通常把计算机系统作为一个复杂的排队系统,用离散状态和不同时刻的瞬时状态(事件)来模拟计算机系统的特征,获得性能评价所需要的参数,例如利用率、吞吐量、响应时间等。

根据描述的方式不同,离散事件模拟分为面向事件、面向活动和面向进程三种方式。面向事件方式是按时序执行事件程序后引起系统状态的变化来实现的。模拟系统内有一个时间控制程序和事件表,事件依发生时刻的顺序排列,模拟系统依次从事件表中选择最早发生的事件,激发该事件,并将模拟时钟推进到该事件发生的时刻,如此依次执行到模拟终止。面向活动方式用活动来表示两个可以区分的事件之间的过程,它标志系统状态的转移,活动是否发生,视规定的事件是否满足激活条件而定,如满足条件则激活相应部件的活动模块。面向进程方式把计算机的系统特性描述成为一组异步的、同时发生并且相互作用着的进程,进程由事件的时间序列及若干活动组成。如排队系统中,从顾客到达系统,直至服务完毕离开系统称为一个进程。事件、活动与进程的关系见图1所示。面向进程方式比较接近实际问题,但工作机制较复杂,而面向事件方式工作简单,易于理解,是现在常用的方法。

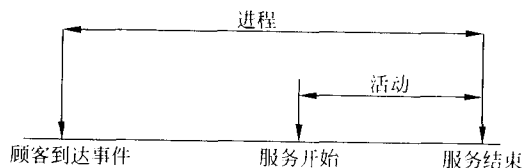


图1 事件、活动、进程的关系

系统模拟一般按以下4个步骤进行:建立模拟模型,编制模拟程序,执行模拟程序,对模拟结果进行统计分析。

结果是模拟工作的重要环节。从分析模拟

结果的观点来看,可分成终止型模拟和稳态型模拟。终止型模拟的运行长度是事先确定的,因为模拟时间有限,系统初始状态影响到系统的性能,所以要求每次运行的初始条件相同。为了消除初始状态的影响,还必须将模型多次独立地运行,其次数应达到一定要求。稳态型模拟的运行长度需足够长,模拟的目的是估计系统的稳态性能。由于运行时间长,初始状态的影响可以忽略,但必须保证有充分的运行长度。根据对置信区间精度的要求,分别采用固定样本长度法、序贯法、均值法、稳态型序贯法、重复产生法及重复删除法进行分析。

由于计算机系统性能模拟的要求,用于编写计算机性能模拟程序的计算机性能模拟语言应当具有如下功能:①对模拟模型有良好的表达能力;②具有系统的动态说明和随机现象描述;③提供模拟运行的控制机构;④为适应不同的模型配置能方便地改变模拟输入参数;⑤可以对模拟过程进行控制;⑥友好的用户界面,较强的人机交互功能;⑦能收集、加工以及显示模拟结果,获得模拟报告。

适用系统模拟的语言有3类:专用模拟语言、通用程序语言和具有模拟功能的通用语言。专用模拟语言为用户提供一种便于表达模型的结构格式,用它编写的模拟器程序功能强,是模拟时最常用的语言。目前这类语言有 GPSS, SIMSCRIPT, SIMULA 等近 30 种。由于它们对计算机系统的模拟采用离散系统模拟,所以,称为离散模拟语言。有些计算机没有提供模拟语言,或用户对模拟语言不熟悉,此时可使用通用程序设计语言,如 FORTRAN, C, PASCAL 等编制模拟程序。通用高级语言具有较好的调试手段和较高的运行效率,但编制程序的工作量较大。可以将通用语言扩充模拟功能,这种语言比模拟语言的编程效率高,兼容性好。但是按照这种语言的语法规则和特征建立起来的模式比专用模拟语言建立的模式可读性差,理解较困难。

GPSS 是用途最广的一种离散模拟语言,主要适用于排队论问题。被模拟系统用流程图描述,各种实体流称为事务,系统的运行用事务在框型间的流通来表示。SIMSCRIPT 是另一种用途广泛的模拟语言,它来源于 FORTRAN,系统的状态由实体及实体的属性来表示,状态由事件来改变。SIMULA 是在 ALGOL 60 基础上发展起来的,建模思想以进程为中心。选择模拟语言时要在分析的基础上注意被模拟系统的特点。因为模拟程序需要的存储量大,运行时间长,也必须考虑计算机的速度和存

储容量。

模拟语言在不断地发展。其发展目标主要是在对建模思想深入研究的基础上使语言更好地模拟客观系统。另外,在模拟语言中引入智能因素,增加逻辑推理、学习机制,以得到智能化模拟语言也是一个主要的发展方向。

模拟语言编写的模拟程序能复现被模拟对象随时间变化的特征,模拟所描述对象的结构和组织,故称为计算机性能模拟器。常用的面向事件的模拟器应包含下列组成部分:系统状态集合、模拟时钟、事件表、主程序、初始化程序、计时程序、事件程序、统计数据保存单元、报告产生程序等。

模拟器工作时必须输入数据进行驱动,由伪随机数发生器来驱动的称为概率模拟或自驱动模拟,由真实系统中测得的数据驱动的称为确定性模拟或称为跟踪驱动模拟。概率模拟是一种蒙特卡罗型模拟方法,根据确定的概率分布,由伪随机数发生器生成作业到达模式、服务时间等随机系列。在模拟前必须对随机序列进行测试以判别它们之间的相关性和序列的变化。确定性模拟的输入序列是从对真实系统进行测量所获得的跟踪数据,由跟踪分析程序进行编辑并转换成适合模拟器输入的形式。确定性模拟不必知道作业或程序的统计特性,模型的验证比较容易,模拟结果更符合实际系统。但在实际系统尚不存在的情况下,则只能采用概率模拟。目前在计算机性能评价中这两种方法均有使用。

在设计模拟器时要考虑以下几个方面:①详细度,用模拟实时时间比和分辨率来表示。模拟实时时间比指在同样的负载情况下模拟器的执行时间与实际系统执行时间之比。分辨率包括时间分辨率与空间分辨率。时间分辨率是被模拟时间的最短值,空间分辨率是模拟器能识别的最小信息。②灵活性,尽量适应系统和工作负载的变化。③选择合适的模拟语言。④决定模型和输入负载。⑤确定输入变量和输出变量。

现在已有为性能评价而设计的专用模拟器,如 CSS, ECSS, ASPOL, PASS, HCSS 等。CSS 模拟器由 IBM 公司设计,它在 GPSS 的结构内部提供更自然的计算机系统描述。ECSS 是 RAND 公司根据 SIMSCRIPT II 研制的语言,它的结构类似 CSS,为用户提供了更大的灵活性,但效率较低。ASPOL 由 CDC 公司设计,内部由 FORTRAN 语言来实现,它以类似 SIMULA 的方式进行描述。还有专为某一类计算机

系统设计的模拟器,如评价数据库工作效率的数据库系统模拟器,用于评价计算机通信网的模拟器等。采用并行处理机进行模拟也在研究之中。

(马范援 郑衍衡)

jisuanji xinxi xitong

计算机信息系统 (computer information system)

利用计算机采集、储存、处理、传输和管理信息,并以人机交互方式提供信息服务的计算机应用系统。通常它涉及的数据量很大,绝大部分数据是持久的、可以为多个应用程序所共享,除具有数据管理基本功能外,还可向用户提供信息检索、统计、事务处理、规划、决策等信息服务。计算机信息系统已广泛应用于各个行业和领域的信息化建设,种类繁多。从功能分类,常见的有电子数据处理、管理信息系统、决策支持系统;从信息资源分类,有联机事务处理系统、地理信息系统、多媒体管理系统;从应用领域分类,有办公自动化系统、军事指挥信息系统、医疗信息系统、民航订票系统、电子商务系统、电子政务系统等。

20 世纪 60 年代以前,计算机主要用于科学计算,计算机应用软件是以数值分析算法为中心展开设计的,数据一般由文件系统管理。后来,出现了一些基于文件系统的计算机事务处理系统,目的是代替人们做某些诸如进、出账管理和自动生成统计报表等事务性操作。70 年代,以数据的集中管理和共享为特征的数据库系统成为数据管理的主要形式,计算机的大量应用也从科学计算开始转向事务处理与分析,包括帮助人们对某些复杂事务进行规划和决策。80 年代以来,随着计算机网络和微型计算机的发展与普及,以信息为中心的计算机信息系统成为主流的计算机应用系统。20 世纪 90 年代以后和进入 21 世纪以来,互联网已成为强大的计算机应用的基础设施,基于网络的计算机信息系统得到广泛应用,从电子邮件、网上聊天、网上游戏、网上购物、远程教学,到企业信息化、政府信息化和军队信息化等,正在为人类的社会进步乃至人们生活方式的改变发挥越来越大的作用。

计算机信息系统可视为面向信息的,由计算机硬件、计算机软件和相关的人员共同组成一个整体的计算机应用系统。计算机信息系统的结构如图 1 所示,通常可以划分为 4 个层次:①基础设施层包括支持计算机信息系统运行的硬件或固件、系统软件和网络;②资源管理层 包括各类结构化、半

结构化和非结构化的数据信息,以及实现信息采集、储存、传输、存取和管理的各种资源管理系统,主要有数据库管理系统、目录服务系统、内容管理系统和元数据管理系统等;③业务逻辑层 由实现各种业务功能、流程、规则、策略等应用业务的一组信息处理代码构成;④应用表现层 其功能是通过人机交互等方式,将业务逻辑和资源紧密结合在一起,并以多媒体等丰富的形式向用户展现信息处理的结果。目前,信息系统的软件体系结构包括客户-服务器和浏览器-服务器两种主流模式,它们都是上述计算机信息系统层次结构的变种。

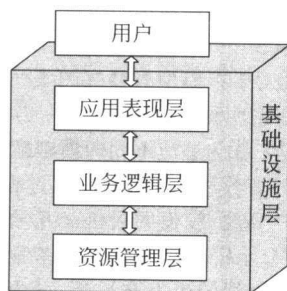


图 1 计算机信息系统层次结构图

当前,计算机信息系统有如下主要特点:①网络化 即计算机信息系统通常都是分布在网络上的,共享的资源不仅包括数据,还包括各种计算资源。②集成化 构建一个计算机信息系统的关键是实现分布在网络上的系统各类资源的有机整合和有效管理。集成的内容主要有基础通信集成、数据集成、应用集成、业务流程集成、门户集成以及企业对企业或部门对部门之间的集成等。中间件是介于网络各结点操作系统之上和应用程序之下的一层支撑软件,已成为开发、部署和运行计算机信息系统,并实现系统有效集成的主流平台。③智能化 随着信息量的指数级增长和计算机信息系统复杂度的不断升级,易用性问题越来越突出,为此在信息系统中融入各种智能技术已成为当前计算机信息系统的发展趋势。目前常用的智能技术包括数据挖掘和知识发现、机器学习、智能搜索引擎和语义 Web,以及为用户提供个性化信息服务的各种技术等。

参考文献

R M Stair, G W Reynolds. Principles of Information Systems. A Managerial Approach, 2003

(吴泉源)

jisuanji xinxi xitong anquan baohu dengji
huafen zhunze

计算机信息系统安全保护等级划分准则 (classified criteria for security protection of computer information system)

中国国家质量技术监督局于1999年发布的计算机信息系统安全保护等级划分的基本准则。它是强制性的国家标准,序号为GB 17859—1999。准则规定了计算机信息系统安全保护能力的五个等级,即:

第一级:用户自主保护级;

第二级:系统审计保护级;

第三级:安全标记保护级;

第四级:结构化保护级;

第五级:访问验证保护级。

用户自主保护级 本级的计算机信息系统可信计算基通过隔离用户与数据,使用户具备自主安全保护的能力。它具有多种形式的控制能力,对用户实施访问控制,即为用户提供可行的手段,保护用户和用户组信息,避免其他用户对数据的非法读、写与破坏。

系统审计保护级 与用户自主保护级相比,本级的计算机信息系统可信计算基实施粒度更细的自主访问控制。它通过登录规程、审计安全性相关事件和隔离资源,使用户对自己的行为负责。

安全标记保护级 本级的计算机信息系统可信计算基具有系统审计保护级的所有功能。此外,还提供有关安全策略模型、数据标记以及主体对客体强制访问控制的非形式化描述;具有准确地标记输出信息的能力;消除通过测试发现的任何错误。

结构化保护级 本级的计算机信息系统可信计算基建立在一个明确定义的形式化安全策略模型之上,它将安全标记保护级系统中的自主和强制访问控制扩展到所有主体与客体。此外,还考虑到隐蔽通道的问题。本级的计算机信息系统可信计算基必须实行结构化是关键保护元素和非关键保护元素。计算机信息系统可信计算基的接口也必须明确定义,使其设计与实现能经受更充分的测试和更完整的复审。本级加强了鉴别机制;支持系统管理员和操作员的职能;提供可信设施管理;增强了配置管理控制;本级系统具有相当的抗渗透能力。

访问验证保护级 本级的计算机系统可信计算基满足访问监控器需求。访问监控器仲裁主体对客体的全部访问。访问监控器本身应是抗篡改的,必

须足够小,能够分析和测试。为了满足访问监控器需求,计算机信息系统可信计算基在其构造时,就应排除那些对实施安全策略来说并非必要的代码;在设计 and 实现时,从系统工程角度将其复杂性降低到最小程度。本级支持安全管理员职能;扩充了审计机制;当发生与安全相关的事件时能发出信号;提供系统恢复机制。本级系统具有很高的抗渗透能力。

参考文献

胡道元等起草. 计算机信息系统安全保护等级划分准则. 国家质量技术监督局,1999 (胡道元)

jisuanji xingneng pingjia

计算机性能评价 (computer performance evaluation)

为了某种目的,选用一定的度量项目,通过实测或通过建立模型对计算机的性能进行测试,并对测试结果做出评价的技术。

计算机性能评价所采用的评价方法大致可分为两类,即测量法和模型法。

测量法通过一定的测量设备或测量程序,测得实际运行的计算机系统的性能指标或有关参数,然后对它们进行统计分析以求出相应的性能指标,这是最直接的性能评价方法。但是这种方法只能适用于已经存在并运行的系统,而且比较费时。

模型法首先对要评价的计算机系统建立一个适当的模型,然后求出模型的性能指标,以便对系统进行评价。此法既可用于已有的系统,也可用于尚未存在的系统,可以比较方便地应用于设计和改进,工作量一般比测量法要小。模型法与测量法是相互联系的,在模型中使用的一些参数,往往来源于对实际系统的测量结果。

下面分别讨论计算机性能评价中的运算速度评价,系统性能指标和性能评测中的基准程序等问题。

运算速度评价

运算速度评价采用计算分析、模拟、测试等方法 and 工具,研究计算机系统执行算术运算和逻辑操作的速度,以便设计、改进和选择具有较高性能价格比的计算机系统。运算速度通常用每秒执行指令的条数或者每秒运行浮点运算的次数来表示,即 MIPS (百万条指令每秒)和 MFLOPS (百万次浮点运算每秒)。它是系统工作能力和生产效率的重要表征。

运算速度评价有以下一些方法:

(1) 吉布森(Gibson)混合法 早期的计算机系

统以算术运算为主要操作,常用加法指令的执行速度来表征机器运算速度。但是,由于计算机系统日趋复杂,仅用加法指令已难以评估机器的运算速度。1959年,吉布森提出从应用程序中统计各类指令所占的百分比,用指令混合比计算指令平均执行速度,称**吉布森混合比**。假设第 i 类指令($i = 1, 2, \dots, n$)在使用过程中出现的概率为 p_i ,其执行时间为 t_i ,则平均执行指令时间 $t_E = \sum p_i t_i$, t_E 的倒数即吉布森混合比平均运算速度。

(2) 数据处理速率(PDR) 美国政府过去为限制高性能计算机出口所设置的运算速度限制性指标,现在已经不再采用。(参见**数据处理速率**)

(3) 综合理论性能(CTP) 美国政府限制高档计算机出口而制定的运算速度限制性指标。CTP以百万次理论运算每秒(MTOPS)表示。(参见**综合理论性能**)

(4) 并行处理系统加速比 程序在单处理机上的执行时间与在由多台处理机构成的并行处理系统上的执行时间之比。加速比可用来描述并行处理的效果。

根据出发点不同,并行处理系统加速比(以下简称加速比)可分为绝对加速比和相对加速比两种。绝对加速比的着眼点放在并行处理相对于串行处理的效果上,以最优串行算法作为比较基准,用它来评价并行算法的好坏,而并行算法不一定基于最优串行算法。相对加速比的着眼点在于并行算法和并行计算机本身的伸缩性。其定义为同一并行算法在单处理机上运行时间与在由多个这种处理机组成的并行处理系统上运行时间之比。

根据加速比的数值范围不同,加速比又可分为超线性加速比、正常加速比和病态加速比3种。超线性加速比是加速比的值大于处理机的个数,通常在两种情况下出现,一种情况是求解搜索问题或NP问题时,多个处理机并行计算,可能很快找到最短求解路径;另一种情况是程序在单处理机上运行时,由于存储容量的限制,处理机频繁地进行空间回收和再分配引起的开销很大,而该程序在多处处理机系统上运行时每台处理机花费在存储管理方面的开销相对减少,甚至无须进行空间回收操作。正常加速比是指在 n 台处理机上的加速比 S_n 满足 $1 \leq S_n \leq n$ 的情况。病态加速比是程序在 n 台处理机上的加速比 $S_n < 1$ 的情况。

根据性能模型不同,加速比又可分为固定负载

加速比、固定时间加速比和固定存储容量加速比。在要求实时响应的应用中,计算工作负载常常是因问题规模固定而固定的。当并行计算机中处理机数量增加时,固定负载就被分配给较多的处理机去并行执行,其主要目的是想尽可能快地得出结果。这种以时间为关键的应用场合所得到的加速比称为固定负载加速比。**Amdahl定律**表明固定负载加速比表示为 $S = \frac{1}{\alpha + \frac{1-\alpha}{n}}$,其中 α 为问题中串行计算部分

所占的比例, n 为并行处理结点数。若 $\alpha = 0$,则最大加速比 $S = n$;若 $\alpha = 1$,则最小加速比 $S = 1$ 。当 $n \rightarrow \infty$ 时,极限加速比 $S \rightarrow 1/\alpha$,即并行处理结点数非常大时,加速比的上限为 $1/\alpha$ 。尽管串行代码所占的百分比很小,但总体性能不可能超过 $1/\alpha$ 。这里的 α 称为程序的串行瓶颈。串行瓶颈问题无法用增加系统的处理机数量来解决。这一结论在过去20年间给人们造成并行处理潜力有限的印象。在以精度为关键的应用领域中,希望像在小机器上解规模小的问题那样在大机器上解规模大的问题,使二者所花的执行时间大体相同。由此提出了固定时间加速比,可表示为 $S = n - \alpha(n - 1)$ 。这表明用增大问题的规模办法使所有处理机保持忙碌状态,在问题扩大到与可用的计算能力匹配时,程序中的串行部分就不再是瓶颈了。也就是说, α 不是个常数,在有些应用场合,当问题规模增大, α 值变小。1993年提出固定存储器容量加速比的基本思想是要在存储空间有限的条件下解尽可能大的问题,这同样需要扩展工作负载,才能有较高的加速比。固定负载加速比和固定时间加速比是固定存储容量加速比的特例。

系统性能指标

系统性能指标是反映计算机系统性能和特征的一组参数。

不同的用户对系统性能指标的关心不同,例如科学计算的用户最关心MIPS、MFLOPS、加速比;军事用户最关心可靠性和环境适应性;过程控制人员最关心实时性和可靠性;维护人员最关心可用性和可维护性。

常用的性能指标有MIPS、MFLOPS、加速比、吞吐率、响应时间、利用率、性能价格比等。通常,性能指标可分为绝对的和相对的两类。绝对性能指标是指不需要一定参照量的指标,如MIPS和MFLOPS。相对性能指标是指相对于一定参照量的指标,如加

速比是多结点并行处理速度相对于单结点处理速度的倍数。性能指标又可分为基本的和导出的。基本性能指标是指不能由其他指标导出的指标,这种指标是直接获得的,如处理时间。导出性能指标是指通过基本性能指标而间接得到的,如 MIPS, MFLOPS 分别是由指令数和浮点操作数与处理时间导出的;加速比是由多结点处理时间和单结点处理时间导出的。性能指标还可分为工作量类、响应性类和利用率 3 种。吞吐率、工作速度、指令执行速率和数据处理速率属工作量类。响应时间、周转时间和反应时间属响应性类。硬件(CPU, 内存, I/O 通道, I/O 设备等)的利用率,操作系统的利用率,公用软件(如编译程序等)的利用率,数据库的利用率属利用率类。

计算机的定点运算速度常用 MIPS 表示。常用的 MIPS 有峰值 MIPS、基准程序 MIPS 和以特定系统为基准而得到的 MIPS 等。对于 1 个处理机芯片或 1 台串行计算机,其峰值 MIPS 值通常是以其指令集中最基本指令的执行速度而计算得到的。如设某一系统的指令集中最基本指令的执行需 a 个机器周期,每一机器周期为 $t \mu\text{s}$, 则其峰值 MIPS 值为 $1/(at)$ 。一台处理机的平均 MIPS 值是指按照一定的加权值平均其指令集中各类指令的执行速度而计算得的 MIPS 值。基准程序 MIPS 值是指用基准程序测得的 MIPS 值。对于某一计算机,所用基准程序的不同,测得的 MIPS 值也不同,这主要是由于不同的基准程序中各种指令的混合比例不同以及指令集中各种指令的执行速度不同而引起的。MIPS 指标用于评价同一厂商生产的同一系列的计算机的定点运算速度比较准确,因为这些计算机有相似的系统结构、操作系统、语言和编译器,尤其是相似的指令集。如果两台计算机的结构和指令集不同,那么用 MIPS 值来比较它们的运算速度是不准确的,有时可能会导致错误的结论。

计算机的浮点运算速度一般用 MFLOPS 表示,它可衡量计算机的科学计算性能。MFLOPS 可分为两种:峰值 MFLOPS 和以基准程序测试得到的 MFLOPS。对于单个处理机芯片或一台串行计算机,其峰值 MFLOPS 通常是以其最快的浮点操作速度或以各浮点操作的平均速度计算得到的。设其最快的浮点操作的执行速度或各浮点操作的平均速度为每个机器周期 a 次,每个机器周期为 $b \mu\text{s}$, 则其峰值 MFLOPS 定义为 a/b MFLOPS。串行计算机的 MFLOPS 值也可用基准程序测得。设基准程序 A 在

其上执行的时间为 $t \mu\text{s}$, 程序 A 中含 F 个浮点操作, 则其 MFLOPS 为 F/t MFLOPS。 F 的计算有不同的方法。有的是直接统计机器所执行的浮点操作次数,有的是根据问题的复杂性分析,从数学模型中计算出来。前者获得的是机器实际执行的浮点操作数,其中可能包括一些辅助性的非问题本身的浮点操作,而后者只包括与问题有关的浮点操作。MFLOPS 可用于比较和评价在同一系统上求解同一问题的不同算法的性能。MFLOPS 还可用于在同一源程序、同一编译器以及相同的优化措施、同样的运行环境下,利用相同的方法对不同系统测试浮点运算速度。

吞吐率指计算机在单位时间内能处理的信息量。例如在事务处理领域中,常用事务处理每秒(TPS)来表示计算机的处理速度,但“事务”没有统一的定义。部分计算机厂商以每秒运算次数(OPS)来表示计算机的处理速度。

响应时间指从给定计算机输入到出现对应的输出之间的时间间隔。响应时间取决于用户输入的信息、系统特性以及在用户输入信息时系统正在处理的其他负载。虽然响应时间是一个随机变量,但在相当长的时间周期内可以用统计规律来描述它。

利用率指在给定的时间内,计算机某一部分软硬件的实际使用时间所占比例。

性能价格比也是常被用作评价计算机的参数之一。它以每元人民币或每一美元能买到多少 MIPS、MFLOPS 或事务处理等来表示,或以每 MIPS 或 MFLOPS 值多少人民币或美元来表示。

必须注意,上述系统性能指标均与工作负载以及系统特性有关,评价者必须清楚是在什么系统和什么样的负载情况下测得的性能指标,否则是没有意义的。

基准程序

用户为测试计算机系统性能所需的一组典型程序称为基准程序。用户可按不同需求使用不同的基准程序,所测得的各种指标可供用户对系统性能的好坏进行判定和比较。基准程序按照应用类型可分为科学计算、商业应用、网络服务、多媒体应用和信号处理等类型。此外,基准测试程序还可以分为宏基准测试程序和微基准测试程序。前者测试的是计算机系统的总体性能,针对某种应用类型,它可对不同系统加以比较,因而对系统采购人员很有用,但它不能揭示系统运行好坏的原因。微基准测试程序测量一个计算机系统的某一特定方面的性能,如 CPU

速度、存储器速度、I/O 速度、操作系统的性能等。目前国际上流行的基准程序可以分为以下 5 类: ①综合型(如 Dhystone, Whetstone 等); ②核心型(如 Livermore Fortran Kernels, NASA 的 NAS 等); ③数学库(如 LINPACK, FFT 等); ④应用型(如 SPEC, Perfect, Splash 等); ⑤并行型(如 NAS 的 NPB, PARKBENCH 等)。

因为计算机系统的飞速发展,测试领域的进一步拓宽,基准程序更新加快,用户需要不断调整所使用的基准程序。

参考文献

1. 陈兴业. 计算机系统性能评价. 广州: 华南工学院出版社, 1987
 2. 李立立, 李亚民. RISC——单发射与多发射体系结构. 北京: 清华大学出版社, 1993
 3. 罗晓沛, 侯炳辉. 系统分析员教程. 北京: 清华大学出版社, 1992
- (吴露成 郑纬民 陈玉霜 詹文岛)

jisuanji yinyue

计算机音乐 (computer music) 利用计算机进行音乐信息处理的技术。计算机具有很强的信息处理能力, 而音乐虽然最终表现为声波的振动, 但声波只是音乐信息的载体, 音乐信息本身则完全可以用计算机来处理。

计算机音乐的历史始于 1956 年。当时美国 Illinois 大学的 L. Hiller 和 L. Issacson 两人在 ILLIAC I 计算机的辅助下创作了 Illiac 弦乐四重奏组曲。此后, 计算机在音乐领域的应用范围不断拓展, 水平也日趋提高。现在, 已有很多商品化的计算机音乐系统上市。

音乐信息在计算机内的表示具有不同的层次, 粗略地可分为信号层、内部表示层和人机界面层。信号层上的音乐信息是符合一定标准(例如电子乐器数字化接口(MIDI)标准)的数字指令, 可以被配有 MIDI 接口的任何电子乐器接受并实时转换成具有听觉效果的声音; 内部表示层上的音乐信息按某种计算机可读的格式储存并接受处理; 人机界面层上的音乐信息则表现为人类可读的乐谱。

计算机音乐的研究内容是多方面的。一些具有代表性的研究分支为:

(1) 乐谱识别 把印在纸上的乐谱通过光电扫描仪转换成数字形式并对之进行模式识别, 最后翻译成内部表示, 以备将来使用。

(2) 乐谱生成(显示或打印) 把音乐的内部表示翻译成人类可读的乐谱, 将其在屏幕上显示出来或付诸打印。其中, 屏幕显示可以是交互式的, 就是说, 计算机可以为作曲家提供一个具有“所见即所得”效果的乐谱编辑器。

(3) 计算机记谱 把音乐的物理实现记录下来并转换成数字形式, 然后根据音色及乐理知识将其分解成多个声部, 经分析后形成相应的内部表示, 最终转换成人类可读的乐谱。

(4) 计算机作曲 包括: ①计算机自动生成音乐动机, 并发展旋律; ②给计算机输入一个音乐动机, 由计算机发展旋律; ③给计算机输入一个单声部旋律, 由计算机编配多声部和声等。要做到这些, 必须把相应的旋律学、和声学、曲式学知识变成计算机可执行的程序——专家系统。

(5) 计算机音乐合成 由计算机、MIDI 键盘、MIDI 合成器、音响等装置可组成方便的音乐创作平台。用户可以在终端前以“所见即所得”的方式编辑乐谱, 并可以随时听到由合成器模拟的各种音色的乐器按乐谱的要求“演奏”该乐谱的整体音乐效果, 对不满意的地方随时进行修改。

(6) 计算机音乐作品分析 通过对特定音程、特定节奏型、特定和声连接以及特定曲式结构的频率统计, 可以分析音乐作品的风格, 包括音乐作品的时代、民族、地域特征及特定作曲家作品的内在一致性和风格变迁等。计算机可为这方面的研究辅助处理大量信息并提供定量的分析数据。

(7) 计算机音乐文档管理 在网络和多媒体技术的支持下, 现在计算机音乐文档管理不仅能提供标题、作者、摘要、乐谱和其他背景材料的检索服务, 还能通过网络实现远程点播服务。在一台具有多媒体播放功能的国际联网计算机上, 可以点播远在千里之外的音乐文档中的任何曲目。

计算机音乐还在随着计算机科学技术的发展而不断发展。随着网络技术、多媒体技术和人工智能技术的进步, 计算机音乐将为职业音乐家和广大音乐爱好者开创无限广阔的音乐新天地。有了计算机的帮助, 音乐家们可以更方便地从事音乐活动, 进入更高的艺术境界。而广大的音乐爱好者和欣赏者们则可以有更多的机会实践自己的音乐理想, 欣赏自己所钟爱的音乐作品。

参考文献

- Howe H. Electronic Music Synthesis. New York: W. W. Norton, 1975

(白頔)

jisuanji yingyong jishu

计算机应用技术 (technologies for computer applications) 计算机在各行各业和各种社会活动中的应用所涉及的基本原理、共性技术与方法。在计算机科学技术发展的初期,计算机的应用只是面向少数领域和部分专业人员,能使用计算机只是一些训练有素的专家。随着计算机科学技术的发展,其应用范围不断扩展,计算机科学技术在系统(硬件和系统软件)及应用两方面有所分工,各有侧重,从而使计算机应用技术有着相对独立的发展。它已成为计算机科学技术中的一个二级学科。国外也称为计算机应用方法学。

计算机应用大体可分为两种类型。一类是数值应用。它包含大量数值计算,处理对象以数学模型和数值数据为主。数值分析、最优化方法、计算几何等是其支持技术,有关的理论、方法已成为数学学科中的一个重要分支,即计算数学(参见**数值计算**),一般不再列入计算机应用技术学科。另一类是非数值应用。它所处理的问题和信息大多表示为符号和规则,其支持技术包括人工智能、语言文字处理、**计算机图形学**、**多媒体技术**、**数据库管理系统**等。

在开发计算机应用时需要运用一系列的原理、技术和方法。对信息进行采集和管理,建立计算机内部模型并对信息进行加工,再以适当的形式表示处理的结果,最终再对社会活动和生产过程进行指导甚至直接进行控制。计算机应用技术涉及上述信息处理过程的全部内容,虽然范围十分广泛,但其共性技术是对信息的处理和管理。这里所说的信息,除数值信息外还包括文字、声音、图形、图像等多种类型。这里所说的信息处理,包括对信息的获取、表示、加工、转换与表现等方面的内容。

语言和文字是人类文化交流的最重要的媒体,它们的处理技术直接影响到计算机在各个领域的应用以及社会信息化的进程。语言文字处理技术是利用计算机对语言文字的音、形、义及其所带信息进行加工、操作的理论、方法和技术。其内容包括语言及文字的自动识别与输入、文档表格的编辑排版与生成、言语的计算机合成以及**自然语言处理**(如**机器翻译**、自动摘要、问题回答等)。而中文信息的处理还包括汉字的编码及输入、**中文信息检索**等一些特殊问题。

图形和图像是信息的另一类主要媒体。它们是计算机图形学和**数字图像处理**的研究对象。计算机

图形学研究如何在计算机内建立真实景物或虚拟景物的模型,并将这些模型转化为可视图像在图形设备上展现。而数字图像处理则研究如何根据应用的需要对图像进行增强、复原、变换、重建等处理,或从中提取关于该图像内容的特征信息。虽然这两门学科涉及的均为图形或图像,其目标或输入输出却正好相反,它们的研究和发展一直属于两个不同的领域,但近十多年来它们互相渗透,结合日趋紧密。

多媒体计算技术是对文本、声音、图形、图像及影视(运动图像)进行综合处理,使它们建立有机联系并集成为交互性很强的信息系统的一门技术,其中声音和影视数字信号以及它们与其他传统信息媒体的综合处理与应用是多媒体技术的核心。

信息管理技术主要研究在计算机中如何有效地组织数据以及如何高效、可靠地储存和检索数据。计算机发展初期采用的是文件系统,随着数据处理应用的发展出现了数据库技术。**数据库**是长期储存在计算机系统内的有组织、可共享的大量数据的集合,管理这些量大、持久和共享数据的核心软件是**数据库管理系统(DBMS)**。目前用于商用信息管理的数据库管理系统已经比较成熟。而用于多媒体信息管理、工程设计制造信息管理等的数据库管理系统,正在研究开发和完善中。

以信息处理和信息管理技术为核心的计算机应用系统,按照它们所提供的服务,可以分成以下几种类型,它们各自具有不同的特性。

计算机信息系统 许多情况下计算机信息系统是一个很笼统的概念,这里特指主要任务是对数据进行采集、储存和处理并以人机交互方式为用户提供信息服务的系统。通常它处理的数据量很大,且绝大部分数据是持久的,可以为多种应用所共享。除了具有数据管理的基本功能外,它还能向用户提供信息检索、统计、事务处理、规划、决策等信息服务。应用最普遍的两种信息系统是事务处理系统和信息储存与检索系统,前者用来处理预先定义的事务并输出相应的处理结果,如订票系统、金融信息系统等;后者用于存储文档类数据,并可向用户提供满足其查询要求的文档全文或摘要,如科技文献检索系统、联机医学文献分析和检索系统 MEDLINE 等。目前计算机信息系统正朝系统集成化、结构分布化、信息多元化、功能智能化、服务公用化的方向发展。

计算机辅助系统 借助计算机在设计、生产、教学等过程中进行有效的辅助性工作,形成一个以人

为主导的人机结合的系统,以充分发挥人的创造力,提高效率,降低成本。目前**计算机辅助技术**已广泛应用于设计、制造、工程以及教育等领域,形成多种不同的计算机辅助系统。虽然计算机辅助技术与各应用领域密切相关,但其主要的辅助功能都是通过以图形为主的人机交互方式来实现的,因此它的发展与计算机图形学有着密切关系。多年来,计算机辅助技术还针对工程数据的管理、交换、规范以及与其他信息系统的互连、互通、互操作等进行研究,使这门技术朝系统化、集成化的方向发展。

计算机仿真系统 计算机仿真是利用计算机建立、校验、运行实际系统的模型以得到其行为特性,从而分析研究该实际系统的方法和技术。这里的“系统”是广义的。它包括工程系统,如电气系统、热力系统、计算机系统等,也包括非工程系统,如交通管理系统、生态系统、经济系统等。随着系统的规模日益庞大,结构日益复杂,仿真技术不但能提高效率,缩短研究开发周期,减少训练时间,不受环境及气候限制,而且对保证系统安全、节约开支、提高质量尤具有突出的功效。

计算机控制系统 计算机控制系统是通过不断采集被控对象的各种状态信息,由计算机按照被控对象的模型和一定的控制策略实时地计算和处理后,作为控制信息去推动执行机构,使被控对象自动地、精确地按照预定的规律运行,以减少人工操作,提高生产效率和产品质量。对于大型复杂系统,还可以借助**计算机网络**的支持,实施既有分散,又有集中的计算机分层控制。**嵌入式系统**是计算机控制系统的一种类型,它特指将计算机嵌入一个物理系统中,用来控制或维护系统中其他组成部分的某种性质或关系,以实现总的系统目标。目前已广泛用于飞机、汽车、家用电器、武器装备、通信设备、医疗设备、玩具等。嵌入式系统通常应满足实时性、对外部事件的响应能力、对机械装置或物理过程的控制能力以及高可靠性等要求,这就给嵌入式系统的软件开发带来了新的挑战。

消息交换系统 计算机网络特别是**因特网**的发展,促使一批新的基于计算机网络的通信应用应运而生。它们统称为消息交换系统。例如**电子邮件**、**IP电话**、**计算机会议**、**计算机支持的协同工作**、**网上聊天**、**网络游戏**、**远程教学**等。以计算机网络为基础的通信比传统的以模拟技术为主的通信有许多优点。它支持多媒体通信,采用同步与异步方式相结合、点对点通信与多播(广播)通信相结合,可实现

智能化的管理与监控等,因而应用形式多样,发展非常迅速。

在各种计算机应用中,系统的友善性和智能化是计算机应用技术追求的两个共同目标。**人机交互技术**和**人工智能**是与此紧密相关的两项关键技术。人机交互技术是依据对人与人、人与系统之间的交互行为的理解,使计算机能够以友善、自然和直观的方式与用户进行信息交换或提供服务的技术。人机交互技术的形成是多学科综合作用的结果。除了计算机科学技术以外,人性因素、人类工程学、工业工程、认知心理学、社会心理学等学科的发展,也对人机交互技术的发展起着重要的作用。人工智能是研究解释和模拟人类智能、智能行为及其规律的一门学科,其主要任务是建立智能信息处理理论,进而设计可以展现人类某些智能行为的计算机系统。

由于**计算机软件**、**计算机硬件**和**计算机应用技术**的发展,计算机应用已逐渐渗透到人类活动的各个领域。它的服务对象已从面向专业人员扩展到面向大众、面向社会。计算机应用系统也由最初的单机系统向网络分布式系统发展。进入21世纪以来,随着计算机及相关技术的进一步发展,通信和计算机设备体积越来越小,价格正变得越来越便宜,各种新型传感器也蓬勃发展。同时,由于人们对生产效率、生活质量的不懈追求,人们开始希望能随时、随地、随意(无困难)地享用计算机能力和信息服务,计算机应用将开始步入一个新时代——普适计算时代。各种新的计算机应用需求,将进一步推动计算机应用技术的发展,而计算机应用技术的发展也将大大促进计算机更广泛的应用,从而加快人类社会信息化的进程。

参考文献

Ralston A, Reilly E D, Hemmendinger D.
Encyclopedia of Computer Science (Fourth edition).
Nature Publishing Group, 2000

(张福炎 何志均 唐泽圣 王能斌)

jisuanji yingjian

计算机硬件 (computer hardware) 构成计算机系统的所有物理元器件、部件、设备、设施以及相应的工作原理、设计与制造、检测等技术的总称。

在计算机问世初期,“计算机”一词实际上只是指“计算机硬件”。进入20世纪60年代,由于程序设计技术的进步,才形成“计算机硬件”和“计算机软件”的概念。

发展历程

由于计算机硬件是计算机的物质体现,所以自从计算机问世以来,其发展都是以硬件的变革作为分代的主要标志。它共分为4代。

第一代计算机以前的计算工具是机械式或机电式的。典型的代表是算盘、手摇计算机和机电式计算机。它们的运算速度慢,精确度差,特别是不能自动进行运算,所以不是现代意义上的计算机。

第一代计算机 (从20世纪40年代中期到50年代末期)是电子管计算机。人们普遍把1946年2月开始运行的ENIAC作为第一台计算机的代表。第一代计算机的运算控制部件使用了大量电子管,不仅运算速度大大提高,而且能实现自动计算。其存储设备最初使用阴极射线管或超声延迟线,以后使用磁鼓存储器和磁心存储器。输入和输出设备则沿袭了当时的机电式高级分类统计装备,如穿孔卡片机、穿孔纸带机和击打式打印机。

由于硬件设备制造成本昂贵,程序长度和数据精度都很有限,计算机的类型和数量也不多,主要用于解决当时较为复杂的科学和工程计算,并且集中于军事方面。

第一代计算机硬件的设计原则是尽量少用电子管。组装工艺采用焊线技术,组装密度低,体积庞大。检测工具主要是电工仪表和示波器。计算机的可靠性差,抗干扰能力低,对机房环境要求高。整机平均故障间隔时间往往不到1小时。

第二代计算机 (从20世纪50年代中后期到60年代中期)是晶体管计算机。1947年晶体管问世,50年代制成晶体管电路。这种电路是第二代计算机的运算和控制部分的主要硬件。与电子管电路比较,晶体管电路具有工作速度快、可靠性高、耗电量少、体积小、成本低廉、适宜大批量生产等突出优点。第二代计算机的存储器曾经使用过多种存储媒体,包括磁膜、磁泡、磁杆等,但它们都只是昙花一现,最后还是稳定在磁心上。第二代计算机的主存储器是磁心存储器。磁鼓存储器和早些时候出现的磁盘存储器用来作为大容量辅助存储器。它扩大了磁心存储器的存储容量。输入和输出设备增加了磁带机和显示器等。

由于晶体管电路的诸多优点,计算机的结构得到较大发展,出现了多种类型的计算机。虽然它们的数量还不是很多,但却开拓了计算机应用的新领域。由过去以军事为主的应用转入以经济为主的应用领域,由以科学与工程计算为主的应用转入科学

与工程计算、事务处理和过程控制等更为广阔的应用领域。

第二代计算机硬件的主要设计原则是提高整机性能,即采用较复杂的逻辑结构,以满足应用对计算机性能和功能的多种要求。计算机制造时采用印制板和绕接连线,从而提高了组装密度,缩小了体积。这就是通常所说的“快(提高运算速度)、大(扩大存储容量)、小(缩小整机体积)”三原则。体积缩小后,散热问题成为工程实施方面研究的重大课题。检测手段采用自检电路(如奇偶检验等),改善了维护条件。抗干扰能力和可靠性也有了提高,平均故障间隔时间可达1000小时以上。

第三代计算机 (从20世纪60年代中期到70年代初期)是集成电路计算机。1958年半导体集成电路问世,1962年有集成电路商品,很快就用于计算机的运算器和控制器。当时的集成电路只能把100个以下的元器件集成在一个芯片上。与第二代计算机相比,第三代计算机大大提高了运算速度,缩小了体积,提高了系统的可靠性。第三代计算机的主存储器仍采用磁心存储器,而作为辅助存储器的磁鼓存储器则逐渐被淘汰,磁盘存储器占据了绝对优势的地位。输入和输出设备变化很大。传统的机电式设备,除击打式打印机外,基本上都被淘汰。磁带机成为重要的输入输出设备。同时,还陆续出现了多种非击打式印刷设备。软磁盘开始被采用作为输入输出设备。由于数据通信的发展,计算机进入网络环境。因此,由电传打字机发展起来的终端设备也成为计算机重要的输入和输出设备。

第三代计算机硬件的主要设计原则是规范化。集成电路的生产宜于数量多而品种少,所以计算机的生产便于实现规范化和自动化。这时,多层印制电路板和高密度组装已普遍用于制造工艺。自检电路的广泛使用不仅促使较为复杂的检测设备(如逻辑分析仪)的出现,而且由计算机自行检测硬件故障的检测程序和诊断程序也日臻完善。计算机对环境的要求不断降低,可靠性指标大幅度提高,以至于用户逐渐淡薄了平均故障间隔时间这一类概念。

第四代计算机 (20世纪70年代中期以来)是大规模和超大规模集成电路计算机。与第三代计算机比较,第四代计算机所用的集成电路芯片在集成度方面虽然只是量的变化(一个芯片上能集成的元器件在10000个以上),但在性能方面却产生了质的飞跃。首先,由一个或几个芯片组成的微处理器,使得计算机的新成员,即微型计算机,进入了人类的

社会生活,从而再一次大规模地开拓了计算机应用的新领域。半导体集成电路技术的发展遵循摩尔定律,其集成度每18个月翻一番。如此快的发展成为微处理器性能突飞猛进的主要动力。其次,半导体存储器取代了延续多年的磁心存储器,并解决了将部分程序固化的问题。半导体存储器的容量越来越大,存储密度按每年60%的速度增长。**辅助存储器**仍以磁盘存储器为主,但发展十分迅速,存储密度按每年60%的速度增长。第三代计算机时期发展起来的各种输入和输出设备继续得到使用和发展。终端设备成为人与计算机交互通信的主要手段。图形、图像和声音的输入和输出设备在技术上逐渐成熟,陆续被采用。

第四代计算机硬件的设计已深入到集成电路芯片之内,设计原则仍然在简化逻辑、实现高性能和保持规范化三者之间求取最佳平衡。制造工艺还是采用多层印制电路板和高密度组装技术,但重点已转到集成电路芯片的制造上了。硬件可靠性进一步提高,检测工作主要由测试软件自动完成。随着计算机一代代的发展,它的设计思想、制造工艺和检测技术在不断地进步,标准化工作也在不断地深入。

综上所述,微电子技术、磁记录技术、光电子技术、精密机械技术、高密度组装加工技术等是促进现代计算机发展的关键,而计算机的发展又对这些技术提出更高的要求。计算机硬件的水平通常反映出同一时期电子工业和精密机械工业的水平。

基本内容

计算机硬件包括所用的**集成电路、计算机逻辑部件、计算机存储设备、计算机输入输出设备、计算机电源、计算机工程设计和制造、计算机硬件测试、计算机硬件可靠性、计算机维护、计算机机房设施**等。现简述如下。

集成电路利用微电子技术将电路中的晶体管、二极管、电阻等元器件以及它们之间的连线制作在半导体芯片上,以完成特定的功能。计算机的各个部件和各种设备都需要使用集成电路,特别是**中央处理器**和存储设备。微处理器可将中央处理器集成在一个芯片上,大量存储单元及其外围电路也可以集成在一个芯片上。半导体存储器芯片分为**随机存取存储器芯片**和**只读存储器芯片**。由于微电子工艺的飞速发展,已可将具有完整功能的系统(包含处理器、局部存储器、接口和专用处理部件等)集成在一个芯片上,称为**片上系统**。另外,还可根据用户的需求设计制作专用芯片。多年来,硅半导体集

成电路是广泛使用的。**砷化镓集成电路、锗硅异质结器件、光电集成电路、超导集成电路**等也处于发展之中。

计算机逻辑部件包括**运算器**和**控制器**两部分。运算器是完成算术运算和逻辑运算的部件。它通常由能完成加法、乘法、除法、布尔代数运算、移位操作等的电路和存放操作数及运算结果的寄存器组成。控制器按照程序规定的顺序,自动接受和执行指令。它解释指令的操作码和生成地址码,并根据译码结果将适当的控制信号送到计算机的有关部件。控制器由**程序计数器、指令寄存器、时序控制部件**和组合逻辑电路组成,这种控制器称为**硬连线控制器**。还有一种控制器是通过执行若干条比指令低一层次的微指令所组成的微程序而完成指令的各种基本操作的,这种控制器称为**微程序控制器**。

计算机存储设备用来存储程序、数据、运算结果和资料等。它可以接受数据和保存数据,并根据命令提供数据。根据功能、结构和工作原理的不同,存储设备可分为**半导体存储器、磁存储器、光存储器、外存储器系统**等。半导体存储器主要用来作为计算机的主存储器,即存放计算机运行时随时需要使用的程序和数据。半导体存储器件也可以做成具有磁盘功能的半导体盘。磁存储器是以硬磁材料为存储媒体的一类辅助存储器,可分为**硬磁盘存储器、软磁盘存储器和磁带存储器**。光存储器也是计算机的一种辅助存储器,它主要是利用半导体激光器而做成的光盘存储器。磁带存储器、软磁盘存储器和光存储器的存储媒体可以脱机保存,易于更换,因此,除了可作辅助存储器外,还可用作输入和输出设备。由于科学技术的发展,需要对数量巨大的数据和资料进行保存和读出,因此出现了外存储器系统。它是由大容量存储设备和控制设备组成的、用以组织和管理数据的存取和传输的辅助存储系统。

计算机输入输出设备把数据输入计算机或把计算机中需要输出的数据传送给用户或其他接收设备。输入输出设备种类繁多,功能广泛。可更换的存储设备,如磁带、软磁盘和光盘等辅助存储器也常用作输入输出设备,它们既可作为输入设备,又可作为输出设备。常用的输入设备有**键盘、鼠标器**,还有**光笔、触屏、数字化仪、光学字符阅读器、条码阅读器、磁卡机、扫描仪、数字摄像头**等。常用的输出设备有**显示器、击打式打印机、非击打式印刷机、绘图机**等。印刷机最初只输出字符,后来能输出图形和图像,可以输出清晰度高和质量极佳的彩色硬拷贝。

汉字可以作为一种特殊图形输出,已成为打印设备普遍具备的功能。通过通信线路或数据传输线路和计算机相连的输入输出设备称为**终端设备**,它不但具有人机交互功能,还可使用户在远离计算机的地方和计算机进行人机交互操作。

计算机电源是为计算机各个部件正常运行提供能源支持的部件。计算机对电源的尺寸、重量和可靠性等指标要求不断提高。要求交流电源能高质量地稳压稳频,要求**直流电源**不仅抗干扰性能强,而且在负载大幅度变化时仍能保持稳定的电压。计算机系统要求不间断地供电,因此需要**不间断电源**,当交流输入电源的变化超出规定范围或当外界电网紧急断电时,它能够及时接通其他电源,或延续供电一段时间,使计算机中的信息不致遗失或遭到破坏。

计算机的工程设计和制造包括**可靠性设计**、**热设计**、**印制板设计和制造**、**集成电路制造**、高速数字信号传输技术、焊接技术、**高密度组装技术**等。由于计算机最主要的硬件是集成电路,其集成度越来越高,功能越来越强,电路越来越复杂,所以集成电路的设计技术是计算机硬件设计的一个主要内容。多层印制电路板的设计和制造,特别是多层布线及连接各层的金属化孔的设计制造是计算机硬件的另一项重要技术。高密度组装应考虑元器件之间的和线间的干扰,以及通风散热等。

计算机硬件测试包括**元器件测试**、**印制板测试**、**电源测试**、打印机和磁盘驱动器等设备的测试、整机测试等。传统的检测工具,如电工仪表和示波器,仍在使用。专门的检测设备,如逻辑分析仪和监测器,已普遍被采用。有些计算机有自检电路,能自动发现一位或多位错误,甚至能自动校正错误。硬件故障可由软件检测,已开发出多种检测程序和诊断程序,可将故障定位在集成电路芯片上。

计算机硬件的可靠性包括**元器件可靠性**、**元器件老化和筛选**、**加固技术**、**防信息泄漏技术**、**电磁兼容性**等。为防止元器件的早期损坏对计算机可靠性产生影响,在组装前要对元器件进行老化和筛选。在对可靠性要求特别高的设备中,可使用容错技术和冗余技术。处于运动载体(车载、舰载、机载或星载)的计算机要有抗震措施。在高温、高湿等恶劣条件下的计算机要有防护措施。为了保密,要防止计算机自身的信息泄漏,还要采取措施防止环境的电磁波干扰。

计算机维护可以保证计算机系统的正常运行。它对计算机系统的运行效率、数据的安全与完整、延

长计算机设备的使用寿命都有重要的作用。

计算机机房设施是保证计算机在机房中能安全可靠地运行而配置的各种设施。特别是大型计算机要求有良好的机房环境,包括适宜的温度和湿度、洁净度、抗静电和电磁干扰、完善的通风系统、不间断电源和良好的接地系统等。

展 望

计算机将在高性能、大容量、小型化、低功耗、智能化等方面发展。未来计算机的功能将会更强,处理速度更快,存储容量更大,性能价格比更高,安全性更好,人机界面更友好,用途更广泛。计算机网络与多媒体技术的发展,不断改变人们的工作方式,提高人们的生活质量。计算机与通信技术的紧密融合,互相渗透,正在加速人类社会信息化的进程。

计算机的主要硬件是集成电路。多年来,集成电路按照摩尔定律发展,即集成电路的集成度(芯片单位面积所能集成的晶体管的数量)每18个月翻一番,性能也相应增加,而价格和功耗却不断下降。在21世纪初期,集成电路芯片上 1mm^2 面积可集成几十万个门电路,一个芯片上可集成上亿个门电路。未来的存储器容量将从10亿(10^9)位发展到 10^4 亿(10^{12})位,集成电路的工作频率将从GHz量级发展到THz量级,数据传输速率将从Gb/s量级发展到Tb/s量级。为了实现Pflops的高性能计算系统,将开发出每秒运算 10^4 亿次的单个芯片。一些专用芯片(如数字信号处理器、图形处理器、计算机网络中的路由器等)的功能将越来越强,处理速度越来越快。

将一个完整的系统集成在单个芯片上的片上系统(SOPC)又是片上系统的发展趋势。由于SOPC的广泛应用,其价格已不断降低。在21世纪初期,具有10层低K介质金属连线的100万个门的SOPC的售价低于20美元,当其工作在100MHz时,功耗约0.27W。包含中央处理器、主存储器、闪存和输入输出端口的低端单片机的价格仅约1美元。

随着微电子技术的进步,除了硅半导体器件继续发展外,锗硅异质结器件、砷化镓器件等也将会得到发展。

除半导体元器件外,光器件、超导器件、量子器件、分子生物器件、纳米器件等正在开发和探索之中。

在信息存储技术方面,半导体存储器、磁存储器和光存储器的容量、存取速度和传输速率可望大幅度提高,而体积、功耗和价格却大幅度下降。一些新

的存储媒体和存储技术也正在研究之中。

计算机输入输出设备正在日新月异地发展。首先,由单一形式发展成多种形式的综合;其次,在某些应用中,由计算机的附属设备发展成主导设备;第三,智能化程度更高。输入输出设备的未来发展将更加符合人体特征。人机交互的输入输出技术和设备将更好地支持文字、图形、图像、声音等多媒体信息的识别和融合,还将支持用户对环境的感知能力,包括多种传感技术、上下文感知技术等。随着移动计算的发展,输入输出设备还可实现无线互连。为了实现可折叠的电子报纸,显示屏幕将是可折叠的,由普通电池供电。

总之,计算机硬件过去以惊人的速度发展,今后的发展更是不可限量的。(夏培肃 张修)

jisuanji yingjian kekaoxing

计算机硬件可靠性 (computer hardware

reliability) 在规定的条件下和规定的时间内,计算机硬件保持其规定功能的能力(或概率)。规定的条件通常包括环境条件、使用条件、维修条件和操作技术。规定的时间是对产品规定的观察时间,包括连续使用、间断使用、储存和一次使用的时间。规定的功能是指产品的各项技术性能指标和失效判据。能力是在规定条件下和规定时间内,完成规定功能的程度。

计算机硬件可靠性包括元器件可靠性、设备可靠性和系统可靠性。可靠性技术包括对可靠性的评价、预测、分配、提高等。对于元器件要运用失效物理学来分析元器件失效的过程,找出这些过程与应力、环境条件和时间等各种因素的相互关系,确定失效模式和失效机理,并提出可靠性保证措施。对于设备和系统要运用概率论和数理统计的理论和方法来研究其故障时间的分布、分布类型和分布参数,分析和预测影响可靠性的因素,提出评价计算机硬件可靠性特征的指标,以及计算、试验和分配方法,并提出计算机系统在设计、制造、试验和使用各阶段的可靠性保证措施。

可靠性技术的发展大致分为4个阶段:①调查研究阶段(1950年—1957年),收集分析元器件的现场数据,研究寿命试验方法,成立专门的可靠性组织。②统计试验阶段(1957年—1962年),研制环境和可靠性试验设备,开展产品统计抽样寿命试验,制定可靠性标准和管理规范,建立可靠性数据收集和交换系统。③可靠性物理研究阶段(1962年—

1967年),分析元器件失效机理,研究高可靠性设计和工艺,探讨加速寿命试验的方法。④可靠性保证阶段(1967年至今),提高整机可靠性,强化可靠性保证、管理、认证制度,形成质量保证体系,发展可靠性试验技术和改进可靠性标准。

根据环境和使用条件的情况,可靠性可以分为:①固有可靠性,指系统在设计、制造时的内在可靠性,包括原材料质量、电路设计的技术水平、机械结构和制造工艺等因素的影响。②使用可靠性,指使用人员和维修人员对系统可靠性的影响,包括使用人员的操作技术水平、维修人员的技术水平及其他各种人为因素。③储存可靠性,指设备或系统处于存放状态时,性能随着保持时间的增加所呈现的各种下降规律。④环境适应性,指系统所处的环境条件对于系统可靠性的影响,包括温度、湿度、振动、冲击、运输、盐雾、霉菌、辐射等。

可靠性及其定量指标是反映产品质量的综合指标,是产品从出厂开始到工作寿命终止全过程的一种特性,因而具有综合性、时间性和统计性的特点。可靠性定量指标有以下几种。

(1) 瞬时失效率 $\lambda(t)$ 产品在 t 时刻后的时间间隔 $[t, t + \Delta t]$ 内失效数与在 t 时刻还在正常工作的产品数之比,称为失效率函数,简称失效率。计算公式为

$$\lambda(t) = \lim_{\Delta t \rightarrow 0} \frac{n(t + \Delta t) - n(t)}{[N - n(t)] \Delta t}$$

式中, N 为产品总数, $n(t)$ 与 $n(t + \Delta t)$ 分别为 t 时刻与 $t + \Delta t$ 时刻的产品失效数, $N - n(t)$ 为 t 时刻还在正常工作的产品数。失效率常用的基本单位是非特(fit), $1 \text{ fit} = 10^{-9} \text{ h}^{-1}$ 。产品典型的失效率曲线呈浴盆状,故又称浴盆曲线。

(2) 可靠度 $R(t)$ 产品在规定的条件和规定的时间内,完成规定功能的概率,其计算式为

$$R(t) = \exp \left[- \int_0^t \lambda(t) dt \right]$$

(3) 不可靠度 $F(t)$ 产品在规定的条件和规定的时间内,丧失功能的概率。也称产品的失效分布或寿命分布或累积失效率,其计算式为

$$F(t) = 1 - R(t)$$

(4) 失效密度函数 $f(t)$ 产品在 t 时刻后的单位时间内的失效概率,其计算式为

$$f(t) = \frac{dF(t)}{dt} = \lambda(t) R(t)$$

(5) 平均寿命 $E(t)$ 对于不可修复的产品,是

指失效前的平均工作时间或储存时间,即平均无故障时间(MTTF);对于可修复的产品,是指两次相邻失效(故障)间的平均工作时间,即平均故障间隔时间(MTBF)。平均寿命的计算公式为

$$E(t) = \int_0^{\infty} t f(t) dt$$

计算机系统的可靠性技术包括避错技术和容错技术。避错技术的目的是尽量减少硬件故障发生的概率,减小系统的失效率。容错技术是利用额外的硬件和时间两种冗余方式掩盖故障的影响。计算机系统从下述三个层次提高系统的可靠性。

(1) 提高元器件的可靠性 指选用高可靠度、高集成度的器件,并对上机的元器件进行可靠性筛选。同时改善环境条件,包括温度、湿度、振动、冲击等物理条件和限额使用,降低元器件的各种应力负载,延长使用寿命。

(2) 提高装置级(插件级或功能部件级)的可靠性 指印制板布线、印制板生产、电气装配等生产管理和质量管理。

(3) 提高系统可靠性 包括冗余结构设计(并行、备用、故障检测、故障屏蔽、动态冗余等),缩短修理时间(自动故障诊断,提高可维性等),系统的自动恢复技术等。

产品的可靠性取决于可靠性设计、元器件优选和筛选以及生产加工工艺三个环节,其中可靠性设计是最重要的。

参考文献

1. 傅佩琛,赵霖,张军英. 计算机系统硬件软件可靠性理论及其应用. 北京:国防工业出版社,1990

2. 猪瀬·博编著. 计算机系统的高可靠性技术. 尤国峻,肖俊远译. 北京:国防工业出版社,1985

(孟凤珍)

jisuanji youxi

计算机游戏 (computer game) 一类用于娱乐、竞争目的的计算机软件。计算机游戏充分使用色彩、图形、动画和声响效果,具有趣味性、竞争性或探索性,为广大计算机用户尤其是青少年所喜爱,优秀的计算机游戏具有较好的休闲、娱乐和教育作用。

早在 20 世纪 70 年代,大学和研究所的一些软件人员就编制了供自己娱乐用的游戏程序(例如 Zork 就是最早的游戏之一)。后来,作为人工智能研究内容的一部分,一些国际象棋和跳棋的模拟程

序也开始开发。随着计算机科学技术的发展,特别是 80 年代微型计算机的出现,计算机游戏得到快速发展,一些专用的游戏机开始出现。专用游戏机有的是手持式,需要与电视机连接使用;有的设计成专用机形式(街式游戏机),使用大屏幕显示,配置了专门的操纵杆、轨迹球、方向盘等,它们都内置了微型计算机,声光效果很好,给用户一种身临其境的感觉。到了 20 世纪 90 年代,随着微型计算机和只读光盘(CD-ROM)存储器的普及,计算机动画、数字音频、数字视频技术的发展,计算机游戏在个人计算机上日益流行,游戏变得越来越精彩、越来越吸引人,市场也越来越大,成为 PC 机软件中的一大门类。其时最流行的游戏有 Doom 和 Myst 等。最近几年,一些计算机游戏开始利用电影制片技术并请电影演员参与,使游戏更像交互式电影。计算机游戏有许多类型,例如策略游戏(如 Black & White 等),体育游戏(如赛车、开飞机、足球比赛等),探险游戏(Myst III: Exile),动作游戏(如俄罗斯方块、积木等),益智游戏(如下棋、打牌、猜谜、拼字等),角色扮演游戏(如三国志)等,还有些游戏可以通过玩游戏来达到教授少年儿童阅读、书写、解题或其他技能的目的,这些游戏称为教育游戏。

计算机游戏大都有吸引人的画面和音响效果,加上生动的故事情节,让人在玩游戏的过程中,体验到惊险、紧张与刺激。计算机游戏的更大魅力在于它具有交互性,玩游戏者可以通过操作,干预和改变游戏的进程或结果,从中体验到在现实社会中感受不到的自身力量和智慧,得到在现实社会得不到的自我肯定及社会肯定,从而获得极大的心理满足。近些年来,在局域网或因特网上进行的网络游戏成为计算机游戏的新宠。网络游戏具有互动性、仿真性和竞技性,玩游戏的人可以在网络的虚拟世界里与不同的玩者(有时可达成百上千人)在同一时刻玩着同一游戏,获得身临其境的逼真娱乐体验,被认为是当今最好玩的游戏。

计算机游戏可以训练人的手脑配合能力,起到开发大脑、提升智力的作用;可以激发钻研、创造的欲望和学习的兴趣。但是,未成年人由于社会认知不足和自我防护意识缺乏,沉湎于计算机游戏,使游戏者生理、心理等方面受到伤害。科学家发现,计算机游戏只刺激了与视觉和运动有关的那部分脑的活动,长期沉迷于计算机游戏,左前脑发育受到伤害后,会进一步影响右脑发育。而且,由于玩游戏时全神贯注,身体始终处于一种姿态,眼睛长时间注视屏

幕,导致视力下降,脖子酸痛,头晕眼花,还会引起植物神经紊乱,体内激素水平失衡,免疫功能降低,引发心血管疾病,胃肠神经官能病,紧张性头疼,焦虑,忧郁等。同时,沉迷于计算机游戏,容易使未成年人减少人际间交流,产生自闭倾向,甚至会患上“电脑自闭症”。更为严重的是,一些以“攻击、战斗、竞争”为主要成分的暴力互动游戏,会使未成年人模糊道德认知,淡化虚拟与现实生活的差异,引发道德失范、行为越轨甚至违法犯罪。

参考文献

Microsoft Corporation. Encarta Encyclopedia Standard 2003. Microsoft Press, 2003 (张福炎)

jisuanji zhengji jiance

计算机整机检测 (computer hardware system testing)

对运行在典型条件下的计算机整机进行的综合性能指标的检查 and 测试。检测的主要内容包括:整机图纸、文档审查,以及系统速度、系统正确性、系统响应能力、系统可靠性、系统可用性、系统可维性、系统稳定性及系统友善性等。进行整机检测的主要目的是对被检测的计算机作出较全面的鉴定和评价,帮助计算机研制、生产厂家改进其产品的性能和质量,帮助计算机用户选购适用的计算机或改进现有计算机的性能。

进行计算机整机检测时,首先,应使被测整机运行在设计技术指标所规定的温度、湿度、大气压强、供电电源、工作频率等环境条件之中。其次,必须选择经过权威计量管理部门鉴定和确认并适用于被测计算机的仪器设备。第三,应依据被测机的特点和用户需求选取或编写并经验证的基准测试程序集和检测用例。最后,要制定详细的检测计划。对在检测过程中可能出现的情况,如因偶然事件(停电、更换操作员等)中断测试的处理、故障类型确认、故障时间判定、故障记录及故障排除等,作出明确的规定。总之要尽可能地保证检测的顺利进行和获得高可信度的检测结果。

整机图纸、文档审查 整机图纸、文档审查是依据被测计算机应当遵从的标准(国标、军标、部标等)检查其图纸、文档是否齐全,是否符合标准,还要通过实际考查被检测计算机的主要技术指标,如工作频率、基本字长,处理器类型、存储器容量、系统结构、实际配置、扩展能力、工作环境条件等,验证是否符合设计要求,从而使检测人员了解被测机器的技术水平,为其在随后的测试中如何掌握标准提供

技术依据。

系统速度测试 系统速度测试有系统峰值速度、系统持续速度和系统估算速度等测试方法。它们各自从不同的角度反映整机系统的速度指标。

系统峰值速度 系统峰值速度 v 是被测系统使用的处理器芯片的标称速度 v_c 同该系统所含执行运算的这种处理器的数量 n 的乘积,亦即

$$v = v_c n \quad (1)$$

系统持续速度 用适合被测系统特点的基准测试程序集,分别在已知标称持续速度的参考机和被测机上运行,测得各个基准测试程序在这两种机器上运行完成所需的时间,而后算出被测机器的系统持续速度,其计算公式为

$$v_a = v_b \sqrt{\prod_{i=1}^n \frac{T_{bi}}{T_{ai}}} \quad (2)$$

式中 v_a ——被测机系统持续速度;

v_b ——参考机标称系统持续速度;

n ——基准测试程序集所含的基准测试程序数;

T_{bi} ——在参考机上运行完成第 i 个基准测试程序所需的时间;

T_{ai} ——在被测机上运行完成第 i 个基准测试程序所需的时间。

系统估算速度 选择被测计算机主要应用领域有代表性的课题集,令其在该被测机上运行,记录各个课题运行完成所需的时间,再根据各个课题的程序执行量,算出平均执行速度,即为系统估算速度,其计算公式为

$$\bar{v} = \frac{n}{\sum_{i=1}^n T_i / q_i} \quad (3)$$

式中 \bar{v} ——系统估算速度;

n ——选用课题集包含的课题数;

T_i ——运行完成第 i 个课题所需的时间;

q_i ——第 i 个课题的程序执行量。

上述三种系统速度测试方法,以系统持续速度测试法最通用,所测得的结果也有较高的可信度和较强的说服力。

整机正确性检测 检测计算机解题的正确性。进行这种检测,首先要依据下述原则选择测试课题:①主要应用领域有代表性的各类课题;②典型算法和常用算法的各类课题;③高效、中效、低效的各类课题。然后,令各个课题在被测机上运行三次,每次运行之间有一定的时间间隔。记录每次运行的结

果,进行比较,若三次运行结果一致并达到被测机设计精度或同已知的正确结果一致,则认为被测机通过了正确性检测。

系统响应能力检测 系统响应能力检测包括系统响应时间 t 和系统周转时间 T 检测。

系统响应时间 系统在交互方式和其设计允许的负荷条件下,自用户终端键入命令的时刻到系统在此终端上输出响应该命令的第一个字符的时间。然而,由于计算机有多种命令,其功能各异,各自的系统响应时间相差较大,所以常用各条命令的系统响应时间的平均值表示。常用的检测方法是在终端上键入 n 条命令,记录每条命令的系统响应时间 t_i ($i=1,2,\dots,n$),然后用式

$$t = \frac{\sum_{i=1}^n t_i}{n} \quad (4)$$

算出系统响应时间。

系统周转时间 系统在批处理方式及其设计允许的负荷条件下,自输入设备将各类作业提交给系统到其各类作业输出结果这段时间的平均值。检测方法是自输入设备将 m 个作业提交给系统,分别记录各个作业的系统周转时间 t_i ($i=1,2,\dots,m$),而后按式

$$T = \frac{\sum_{i=1}^m t_i}{m} \quad (5)$$

算出系统周转时间。

系统可靠性检测 系统可靠性常用系统的平均故障间隔时间 (MTBF) 表示。目前常用的检测方法是:根据被测计算机的设计所允许的负荷能力,选用适当数量的能够反映被测机特点及性能的正确性测试课题,使其同该机的综合诊断程序一起按某种方式在被测机上联合运行。运行时间至少应为被测机的 MTBF 设计指标的 3~5 倍的时间。其运行方式通常是正确性课题运行时间占整个运行时间的二分之一以上,而综合诊断程序运行时间占二分之一以下。在运行过程中详细记录系统故障、故障次数及修复时间等,运行达到确定的时间后,作出正确性报告和故障报告。在运算正确的条件下,算出被测整机的 MTBF,其计算公式为

$$MTBF = \frac{T_R}{K_S + 1} \quad (6)$$

式中 T_R ——被测整机正确运行的时间总和;

K_S ——被测整机在被测期间的系统故障

次数。

系统可用性检测 系统可用性常用系统可用度 A 表示。系统可用度的测试方法同系统可靠性测试方法一样,可以使用在系统可靠性检测中获得的记录,算出系统的可用度,其计算公式为

$$A = \frac{T_R}{T_R + T_S} \quad (7)$$

式中 T_R ——被测整机正确运行的时间总和;

T_S ——系统故障修复时间总和。

对于多机或多处理器或有多重部件的计算机系统,还可进行降级使用能力检测。它包括:①在被测系统停机时,对其可能降级使用的多种环境分别进行分割组合,然后令其运行测试课题,检验运行结果的正确性;②在系统运行状态下,通过人机通信切除或连接系统的某个部件,观察系统运行的变化和检验运行结果的正确性;③在被测系统运行状态下,人为地设置故障,由系统自行检出并切除故障部件,继续运行,完成后,检查运行结果的正确性。总之,必须在运行正确的前提下确认被测系统的降级使用能力。

系统可维性检测 系统可维性常用平均修复时间 (MTTR) 表示。故障修复时间取决于维修人员技术熟练程度、维修的方便性、同类部件的互换性程度以及用系统的综合诊断程序进行故障检出和故障定位的速度及精度水平。这些影响系统可维性的因素最终都体现在平均修复时间中,因而可以使用在系统可靠性检测中获得的数据算出被测整机的平均修复时间,其计算公式为

$$MTTR = \frac{\sum_{i=1}^{K_S} t_i}{K_S} \quad (8)$$

式中 t_i ——第 i 次故障的修复时间;

K_S ——被测系统在测试期间的故障次数。

系统稳定性检测 计算机整机稳定性检测是计算机在其设计的环境指标的极限条件下能够正常运行程度的检测。常用的检测方法是对被测整机按某种标准施以各种环境极限条件,如振动、温度、湿度、电源拉偏、工作频率拉偏或电磁干扰等,令被测机在一段时间内运行基准测试程序或综合诊断程序。检验运行结果是否正确,并由此判断系统稳定性是否达到设计目标。

对微型和中型计算机的稳定性检测可以按照 GB 9813—88 和 GB/T 137323—9 中的规定及其所引用的有关国家标准设置环境极限条件。对大型或

巨型计算机的稳定性检测,目前尚无标准可循,常用的耐振能力测试是将整机拆开装车,在三级公路上运输 250 km 作为施振条件。对电源、工作频率常提供 $\pm 5\%$ 的拉偏范围。对抗电磁干扰能力的测试,可按国家标准 GB 9254、GB 6833 的有关规定提供环境极限条件进行。

系统友善性检测 系统友善性是计算机产品获得用户的重要条件之一。主要内容有系统使用方便程度、对误操作的系统防护和报警能力、求助系统的良好程度及可视性程度等。常用的检测方法是:先审查被测机有关系统友善设计的指标,同当时公认的友善性良好的机型相比,判别该机友善性设计完善程度。而后进行现场检测,通过实际编程、调试检验使用的方便程度;通过人为操作错误检验系统自身防护能力和报警能力;通过逐项求助验证求助系统的完善程度。然后通过荧光屏检验用户能直接看到系统的什么状态,如资源忙碌程度、作业调度情况、课题运行过程等。最后作出被测机的友善性评价。

以上检测内容使用了简易的采样检测方法和在计算机研制、调试、维修中常用的仪器作为检测工具。在大部分项目检测中,采用当前使用最多的综合基准测试程序集作为被测机的工作负荷。如此检测得到的结果能够对被测整机作出较全面的和较准确的评价。多年来人们为研究出具有普遍性和代表性的用于计算机检测工作负荷、事件驱动的软件、硬件监测器等检测工具作了大量的工作,取得了显著的进展。然而,研究出一种广泛适用的计算机检测标准规范,还需计算机设计、制造、应用等领域的人们共同努力。

参考文献

1. Domenico F et al. 著. 计算机系统的测量和优化. 郑衍衡, 王春元等译. 北京: 水利电力出版社, 1988
2. 吴立德, 吴霁成. 计算机系统性能评价. 上海: 上海科学技术出版社, 1986
3. 傅佩琛, 赵霖, 张军英. 计算机系统硬件和软件可靠性理论及其应用. 北京: 国防工业出版社, 1990

(程乐祥)

jisuanji zhichi de xietong gongzuo

计算机支持的协同工作 (computer supported cooperative work, CSCW) 地域分散的群体借助计算机及其网络技术的支持相互协调与协作来完成一项任务的技术领域。它包括协同工作体

系结构、群体协作方式、模型和支持群体工作的相关技术、应用系统的开发等部分。通过建立协同工作的环境,改善人们进行信息交流的方式,消除或减少人们在时间和空间上的相互分隔的障碍,节省工作人员的时间和精力,提高群体工作质量和效率,从而提高企业、机关、团体,乃至整个社会的整体效益和人类的生活质量。

计算机技术的发展促进了人类社会的信息化。随着信息化进程的深入,通信技术与计算机及其网络技术相融合,产生了一个新兴的研究领域——计算机支持的协同工作,简称**计算机协同工作**。它是协同科学在社会信息化进程中发展的一种必然产物。它将提高人们的工作效率,促进社会生产力的发展,势必深刻影响人类群体生产方式、工作方式和生活方式。

计算机协同工作 CSCW 的概念是 1984 年被正式提出来的。它是一个多学科交叉的研究领域。在实际应用中,计算机网络与通信技术、多媒体技术等计算机技术的支持,以及社会学、心理学、管理科学等多种领域学者共同协作,向人们提供了一种全新的工作环境和交流方式。

计算机支持的协同工作的分类

群体协作方式的多样性,为 CSCW 研究提供了丰富的内容。在 CSCW 系统中,人们围绕着共同的任务需要进行交互通信、协调、协作和协同等基本活动,可以根据 CSCW 系统中的基本活动方式、群体成员地理分布位置、使用的基本工具和工作环境、应用等对 CSCW 系统进行分类:

按群体成员之间的交互协作方式可分为同步方式和异步方式两种。

按群体成员的地理分布可分为同地协作和异地或远程协作。

按群体规模可分为两人协同系统和多人协同系统。

按使用的基本工具和工作环境可分为信报系统,即**电子邮件系统**、**公告板系统**、**会议系统**、**协同写作和讨论(编著)系统**、 **workflow 系统和群件等**。

按 CSCW 应用系统可分为**协同科研系统**、**协同设计系统**、**远程医疗系统**、**远程教育系统**、**协同决策系统**、**军事协同(参谋会议)系统**和**协同办公系统**等。

按照上述各种分类的观点,可以把各种 CSCW 系统构成如图 1 所示的一种立体模型。

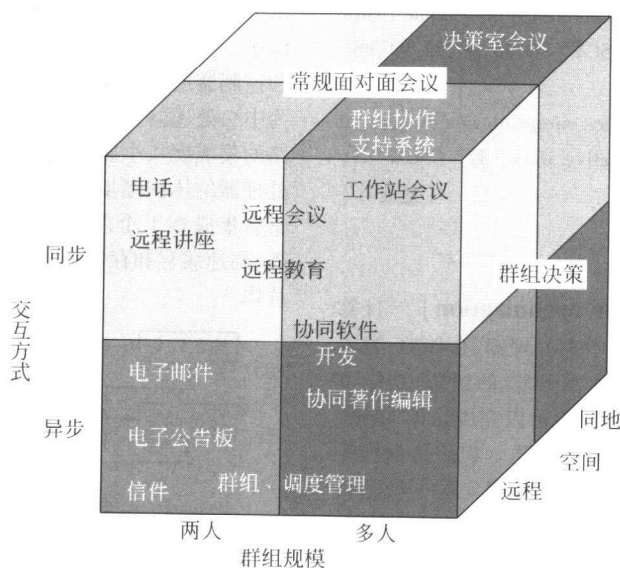


图1 计算机支持的协同工作(CSCW)基本系统的分类

CSCW 系统体系结构

计算机协同工作(CSCW)系统典型的体系结构可以表示为如图2所示的一个4层结构的模型。第1层为“开放系统互连环境”，提供开放的通信网络支持环境，保证协同工作过程中有效的信息交流。第2层为“协同工作支撑平台”，解决协同工作所需的主要机制和工具。主要机制如信息共享、信息安全控制、群体成员管理，基本工具包括电子邮件系统、会议系统、协同写作和讨论系统、工作流系统等。第3层为“协同工作应用接口”，在这一层中需要提供协同应用的编程接口(API)、人机交互(HCI)和智能外围接口(IPI)。通过标准化的服务接口向应用系统提供第3层的功能，使上层的应用系统与下层的支撑平台具有相对的独立性；提供有效、灵活、方便的人机交互接口，以及在协同工作环境下协作

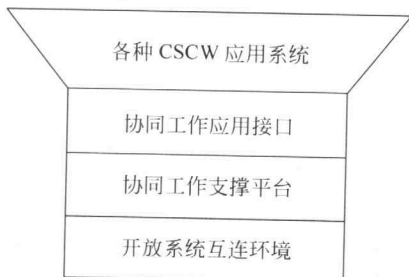


图2 计算机支持的协同工作(CSCW)系统体系结构

各方交互关系，规则和策略等。第4层为“各种CSCW应用系统”，针对各种协同工作应用领域，提供所需的协作支持工具的剪裁和集成，以及协同应用系统的开发。

计算机协同工作的关键技术

CSCW 主要技术基础是计算机及计算机网络技术，主要的发展动力来源于广泛的应用需求。关键技术的深入研究是CSCW应用系统飞跃发展的基础。它们包括：CSCW系统模型和体系结构、群体协作模式、协作控制机制、CSCW系统中的群组通信支持、多媒体技术、多重交互接口（人际交互、人机接口、应用编程接口）技术、应用共享技术、CSCW系统的安全控制、CSCW应用系统开发环境和应用系统集成技术等。

计算机协同工作技术的应用领域

CSCW 具有十分广泛的应用领域，例如，军事应用、工业应用、协同计算机辅助设计(CO-CAD)、办公自动化(OA)和管理信息系统(MIS)、远程医疗、远程教育、合作科学研究、电子商务和商业、贸易、金融领域中的应用、电子政务和各级政府部门协调和决策支持等，几乎包括了人类社会生活、生产的所有领域。

参考文献

1. 史美林. 计算机支持的协同工作：理论和应用. 北京：电子工业出版社，2000

2. Proceedings of the conference on computer supported cooperative work: CSCW'86,88,90,92,94,96,98,2000

3. Proceedings of the european conference on computer supported cooperative work: E - CSCW'89,91,93,95,97,99 (史美林)

jisuanji zucheng

计算机组成 (computer organization) 计算机的各个主要功能部件(中央处理器、存储器、输入输出设备以及各部件之间的互连)的组成、功能及其实现。计算机通过执行程序可以完成计算、模拟、控制、辅助设计和事务处理等工作。程序是由指令组成的,因此计算机执行程序的过程实际上就是按照一定的次序执行一系列指令的过程。计算机从输入设备接收程序和数据,存放在存储器中,在中央处理器的控制和参与下处理数据,完成每条指令的操作,最后将结果数据通过输出设备输出。

计算机的基本组成

1945 年冯·诺依曼等人提出存储程序的计算机方案,其后各国研制的计算机基本上采用这种方案,称为冯·诺依曼计算机或存储程序计算机,直到今天,虽然计算机体系结构已有很大的发展,但仍可认为大部分计算机没有违背冯·诺依曼计算机的基本原理。早期的冯·诺依曼计算机的主要特点为:①采用存储程序方式,程序和数据存在同一存储器中。②存储器结构是按地址访问的线性编址的一维结构,它的字长是固定的。③数据以二进制表示。④机器以运算器为中心,输入输出设备和存储器之间的数据传送都经过运算器。⑤指令由操作码和地址码组成,一般按它在存储器中的存放顺序执行,程序的分支由转移指令实现。图 1 是以运算器为中心的计算机组织。这种由运算器、控制器包揽一切指令操作和输入输出操作的组成,使计算机必须等待正在执行的输入输出操作完成后才能进行下一操作,影响了计算机的运行效率,代之而起的是以存储器为中心的计算机组成如图 2 所示。批量的输入输

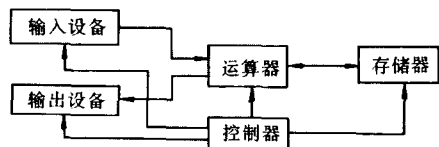


图1 以运算器为中心的计算机组成

出数据可以直接在输入输出设备和存储器之间进行传送。只是在必要时才用中断处理方式请求运算器 and 控制器进行干预。后来将运算器和控制器统一称为中央处理器。随着元器件的改进,尤其是集成电路的集成度从小规模逐步发展到超大规模,使中央处理器的体积逐步从几个机柜缩小到 1 个插件板,最后集成在 1 个芯片内。现在甚至可将浮点处理器、高速缓存和存储控制部件集成在中央处理器芯片内。

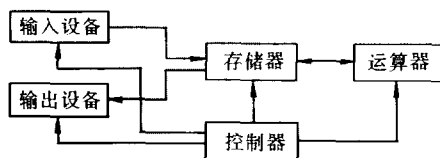


图2 以存储器为中心的计算机组成

下面分别介绍计算机的基本组成部件。

中央处理器 CPU 中央处理器包括运算器和控制器。

运算器 运算器至少包括:①完成定点数的算术运算和逻辑运算的算术逻辑运算单元 ALU(参见**算术逻辑部件**);②完成移位功能的移位器(也可以在 ALU 中完成移位操作);③提供操作数或保存中间运算结果的通用寄存器或专用寄存器。

ALU 一般不包括独立的乘法器和除法器,乘法和除法运算可通过多次交替执行加法(或减法)运算和移位操作来实现,早期的小型计算机和微型计算机甚至连这种功能也没有,而由子程序来完成乘法和除法运算。功能较强的中央处理器内可设置专门的高速乘法器和除法器,还允许它与 ALU 并行工作,以提高计算机的运算速度。浮点运算可根据计算机应用的要求,用程序实现或由浮点运算部件实现。过去浮点部件不包括在 CPU 芯片内,而有专门的浮点运算芯片,但随着器件集成度的提高,到 20 世纪 80 年代末和 90 年代初,在 CPU 芯片内可包括浮点运算部件。为了提高运算速度,在中央处理器内除了设置多个能同时并行工作的功能部件以外,每个功能部件还可以采取流水线方式工作。

早期的计算机执行算术逻辑运算指令时,其中一个操作数往往由运算器中的累加器提供,另一操作数在存储器中,运算结果也放在累加器中。后来的计算机一般设置多个**通用寄存器**,它们既可提供操作数和保存操作结果,也可提供各种**寻址方式**所需的基址、变址、主存地址,或保存调用程序的参量

和堆栈指针等。除此之外,还可设置诸如暂存寄存器、条件码寄存器等专用寄存器。RISC 计算机内设置有数量相当多的通用寄存器,这种计算机只有存取取数指令才访问存储器,其他操作都在寄存器之间进行,因此在采用流水线组成后,使得大多数指令能在一个机器周期内完成。

堆栈计算机用存储器中的堆栈来替代通用寄存器,但其栈顶和靠近栈顶的一些单元内容可能用寄存器存放,对于一般的双操作数算术逻辑运算,采用零地址指令对栈顶的两个操作数进行运算,结果送回堆栈。中央处理器内还设置一些寄存器用来保存主存中程序区、数据区和堆栈的各种指针。

控制器 中央处理器中的控制器主要是指控制指令流和控制每条指令执行的指令控制器,由它产生执行各类指令操作所需的时序控制信号。这些控制信号可以用硬连线逻辑(又叫组合逻辑)(参见硬连线控制器)产生,也可以用微程序控制方式(又叫存储逻辑)(参见微程序控制器)产生。两者相比,前者能获得更快的指令执行速度,后者更适宜于复杂指令的控制信号的设计。有些计算机控制器是由这两种方式结合构成的。为了获得尽可能高的指令吞吐率,可以采用指令重叠执行的流水线技术;或采用先行控制的方法提前取出指令进行预处理,当遇到执行时间较长的指令时,尽可能把条件具备的后续指令提前执行;也可以设立指令缓冲站,以减少从主存储器重复读取同一指令的时间。

中央处理器除完成运算控制和指令控制功能外,还可通过中断系统调用操作系统,对存储器系统和输入输出设备进行统一管理;也可通过中断系统实现实时处理。

存储器 一般指的是存储程序和数据的动态随机存储器(DRAM),也称主存储器。中央处理器可直接访问它,外围设备也频繁地和它交换数据。存储器的存取速度往往满足不了中央处理器的快速要求,它的容量也满足不了用户解题的需要,为此提出了多层次存储系统和虚拟存储器的概念。存储系统可分为3个层次,它们是:高速缓冲存储器 cache(简称高速缓存)(参见高速缓冲存储器)、主存储器和辅助存储器。为了提高存储器的存取速度,在主存储器与中央处理器之间增加了高速缓存;为了扩大存储器容量,使用磁盘存储器等作为辅助存储器(主存储器的后援)。用户使用时,可以面向辅助存储器的逻辑地址空间。从而使存储器达到高速(接近于高速缓存的速度)、大容量(接近于辅助存储器

的容量)的要求。数据和指令在高速缓存与主存储器之间调动是由硬件自动完成的;在主存储器与辅助存储器之间调动是在硬件和操作系统控制下自动完成的。

在某些计算机的存储器中还包含少量只读存储器,用来存放一些操作所必需的基本程序,例如引导程序和基本输入输出系统 BIOS 等。

存储控制器实现对存储器的读写控制,它接收来自中央处理器或输入输出设备、辅助存储器的读写请求,向存储器发读写命令。并对同时来的读写请求,根据优先级进行仲裁。另外,对动态存储器还要控制进行“刷新”。

外围设备及其控制 磁盘存储器、磁带存储器等称为辅助存储器。辅助存储器和输入输出设备统称为外围设备。低速的外围设备通过中断系统与主机(中央处理器和存储器)交换数据。高速外围设备直接与存储器交换成批数据,称为直接存储器存取 DMA(参见直接存储器存取),但在一批数据传送前或传送结束后或传送过程中产生故障时,仍然需要通过中断程序予以处理。

在外围设备数量较多的情况下,全部设备都与存储器分别独立相连是困难的,而且也不能适应进一步扩充数量和更换品种的需要。为了适应计算机系统灵活多变的要求,现代计算机普遍采用总线互连方式,将所有的外围设备经过统一的总线(称为系统总线或输入输出总线)和中央处理器、存储器相连接,如图3所示。所有的外围设备都经过各自的控制器(称为接口)连接到总线上。为了增强通

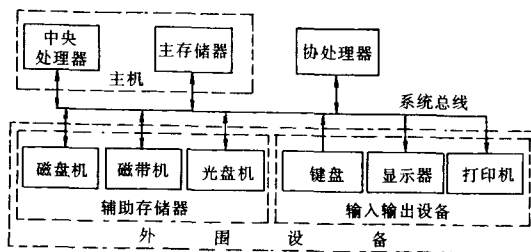


图3 以总线互连的计算机组成

(图中的协处理器可以是浮点部件、数组处理机或图形处理加速器等)

用性,总线的标准化工作很重要,例如 ISA, EISA, Multibus, VME, PCI, NuBus 和 Future Bus 等均为标准总线。由于各部件间传送信息都是通过总线进行的,而在同一时间内,一组总线只能为一对功能部件

服务,因此当同时存在两个或两个以上的传送要求时,需要根据优先级进行排队,因而总线有可能成为系统的瓶颈。在设备数量多和速度要求高的情况下还有其他连接方式,例如在主机内增设高速总线,使得中央处理器与主存储器之间的传送通过高速总线进行;或者设置通道处理机(或输入输出处理机)将所有的外围设备管起来,分担主机的部分工作。

计算机中常用的逻辑电路

中央处理器、存储控制器和输入输出接口部件基本上是由各种逻辑电路构成的。在计算机中常用的逻辑电路可分为组合逻辑电路和时序逻辑电路两大类。

组合逻辑电路 这种电路和输出状态仅和当时的输入状态有关,而与过去的输入状态无关。常用的有加法器、译码器和数据选择器等。构成这些电路的基本单元是“与”门、“或”门和反相器,通常可用逻辑表达式(或用真值表)来表示其功能。

加法器 一位加法器可对两个相加数中的某一位以及从低位来的进位数相加,产生本位运算结果以及向高位的进位数。32个加法器可实现两个32位数的加法运算,但进位数延时较长,因此通常附加有快速进位电路。加法器是构成运算器的主要单元。

译码器 译码器有 n 个输入变量, 2^n 个输出信号,当输入为某一组值时,仅有对应的一个输出为“1”,其余均为“0”的信号。译码器在计算机中得到广泛的应用,如指令译码器、存储器中的地址译码器等。

数据选择器 从多个输入通道中选择一个通道的输入数据作为输出数据。

时序逻辑电路 这种电路的输出状态不但和当时的输入状态有关,而且还与在此之前的输入状态有关。时序逻辑电路内必须要有能存储信息的元件(称为触发器)。触发器是构成时序电路的基本单元,用触发器和控制门可组成寄存器和计数器等,在计算机中应用的有指令寄存器、数据寄存器、缓冲寄存器、移位寄存器和程序计数器等。

另外,由基本门电路组合而成的可编程逻辑器件PLD在计算机中经常被采用,PLD由两级门阵列电路构成,第一级是“与”阵列,第二级是“或”阵列,根据“与”阵列和“或”阵列是否可由使用者编程又有可编程只读存储器PROM、可编程逻辑阵列PLA、可编程阵列逻辑PAL、通用阵列逻辑GAL和现场可编程门阵列FPGA(参见专用逻辑集成电路)等。

现代计算机组成特点

单处理器向提高主频和增强功能方向发展 由于芯片集成度和速度的提高,现代微型计算机几乎已具有过去大型计算机的所有特点:

运算部件的多功能化 微处理器芯片内可含有多个ALU部件、乘法器、除法器、地址部件(形成访存地址)和浮点部件等。

流水线 包括多条指令重叠工作的指令流水线和多个数据重叠处理的运算流水线。指令相关、数据相关、转移指令和中断处理会影响流水线效率,为此可采用编译优化、数据旁路、转移预测等方法进行改进。

二级高速缓存 微处理器芯片内含有几KB到几十KB的第一级指令高速缓存和数据高速缓存(共用或分开),片外可根据需要连接容量更大的第二级高速缓存。某些微处理芯片甚至考虑了三级高速缓存方案。

虚拟存储器 设置存储管理部件MMU,将逻辑地址转换成物理地址,并与操作系统配合,自动完成主存储器与辅助存储器之间的数据传送。有的微处理器已将原来在芯片外的存储管理部件移到片内。

多处理器 微处理器芯片内有适应多个处理器一起工作的指令和相应逻辑。

接口标准化 使系统便于升级。

单机向多机并行处理系统发展 现代微处理器芯片的主频将很快接近其极限,因此依靠提高主频来提高机器性能的潜力已不很大,更有效地提高性能的途径是改进系统结构,提高并行度。

在科学研究和工程设计的某些应用领域,需要对巨大的数组进行计算,一般的通用大型机不适合于这种大量的数值计算,为此,发展了向量流水线计算机(称为向量处理机)。

阵列处理机是一种并行处理系统,它有1个指令部件和多个数据处理单元。在指令部件控制下,所有的数据处理单元对分配来的数据并行地完成1条指令所规定的操作。

多处理机系统(参见并行处理系统)将很多个处理器组合在一起构成一个计算机系统,速度可以很高,价格相对来说比较便宜。

集中式处理系统向分布式处理系统发展 将地点分散的、具有独立工作能力的多台计算机通过通信网络相互连接起来,在系统软件控制下,实现资源共享并共同完成一个任务的系统,称为分布式计算机系统。连接计算机的通信网络可以是局域网(参

见局域网)、城域网(参见城域网)、广域网(参见广域网)或互联网(参见互联网)。所连接的计算机可以是同种机型的或异种机型的(参见分布式异构型处理机系统)。客户-服务器系统(参见客户-服务器器计算)、计算机簇(参见计算机簇)等都是分布式计算机系统。

开放系统(参见开放系统) 开放系统指计算机体系结构、系统总线、操作系统、窗口系统、数据库、图形用户接口、计算机网络和通信服务等都是开放的,符合与制造商无关的国际标准或工业标准。这样,硬件和软件厂商就很容易进行分工,他们的产品也能很方便地集成在一起工作。用户可以选用市场上最适合于他们的软件和硬件组成计算机系统。因此具有开放系统特点的计算机系统之间有着良好的可移植性和互操作性。可移植性是指在一个计算机系统上运行的操作系统(或应用软件)可以很容易移植到另一计算机系统上。互操作性是指不同厂家的各类计算机可以相互交换信息,为达到互操作性要求,各厂家的计算机应符合统一的通信协议、数据格式等,而且具有良好的互连性。

参考文献

1. 王爱英主编. 计算机组成与结构. 第三版. 北京: 清华大学出版社, 2001
2. 孙强南, 孙昱东. 计算机系统结构. 北京: 科学出版社, 1992 (王爱英)

jisuan jihe

计算几何 (computational geometry) 研究几何外形信息的计算机表示、分析和综合的新兴边缘学科。它是计算机辅助几何设计(CAGD)的数学基础。20世纪60年代起,计算几何的产生及最初应用,主要是围绕航空、造船和汽车三大工业部门的产品几何外形设计。现在,计算几何的应用已扩展到许多技术领域,例如:飞机模拟训练、石油勘探地层结构图重建、服装设计、动画片和艺术图案的创作及地形图复原等。计算几何目前研究的主要内容有:贝济埃曲线和曲面, B样条曲线和曲面, 孔斯曲面等。贝济埃曲线是由法国雷诺汽车公司工程师Bézier于1962年提出的,当时主要用于曲线和曲面的构造。后来,该公司以这种曲线为基础发展了一种自由型曲线曲面的设计制造系统,称之为UNISURF系统,并于1972年正式投入使用。

贝济埃曲线 n 次贝济埃曲线的数学表达式是 n 次参数曲线段

$$\mathbf{r}(t) = \sum_{i=0}^n B_{i,n}(t) \mathbf{r}_i \quad (0 \leq t \leq 1) \quad (1)$$

式中 $\{B_{i,n}(t)\}$ 是 n 次伯恩斯坦基函数族,其表达式为

$$B_{i,n}(t) = C_n^i t^i (1-t)^{n-i} \quad (i = 0, 1, \dots, n)$$

$n+1$ 个矢量 $\{\mathbf{r}_i\}$ 是由设计者事先给定的,它们的终点组成了贝济埃控制多边形的顶点(简称贝济埃点)。

贝济埃曲线最明显的优点是:它以 \mathbf{r}_0 为起点, \mathbf{r}_n 为终点,在 \mathbf{r}_0 处与控制多边形的第一条边相切,在 \mathbf{r}_n 处与控制多边形的最后一条边相切;贝济埃曲线在其所有控制顶点形成的凸包之中。贝济埃曲线与它的控制多边形的关系可以这样认为:控制多边形是贝济埃曲线的大致形状的勾画,而贝济埃曲线是对控制多边形的逼近。在设计曲线时,通过人机交互方式,适当地移动控制顶点的位置可灵活地控制贝济埃曲线的形状,从而达到设计者满意的效果,工程中常用的是三次贝济埃曲线。图1给出了两个不同的控制多边形对应的三次贝济埃曲线。

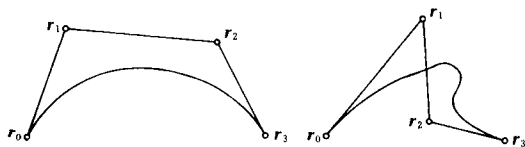


图1 三次贝济埃曲线

有理贝济埃曲线是为了统一表示自由曲线和圆锥曲线而发展起来的,它也是一种逼近曲线。通过对贝济埃点的调整来控制曲线的形状,同时每个贝济埃点对应的权因子又为形状控制提供了新的自由度。 n 次有理贝济埃曲线的数学表达式为

$$\mathbf{r}(t) = \frac{\sum_{i=0}^n \omega_i B_{i,n}(t) \mathbf{r}_i}{\sum_{i=0}^n \omega_i B_{i,n}(t)} \quad (0 \leq t \leq 1) \quad (2)$$

式中 $\{\omega_i\}$ ($\omega_i > 0$) 是权因子,其他与式(1)相同。特别当所有权因子 ω_i 为一非零常数时, n 次有理贝济埃曲线(2)退化为 n 次贝济埃曲线(1)。

贝济埃曲面 贝济埃曲面是两组伯恩斯坦基函数通过张量积拓广成的参数曲面,也可以看成是贝济埃曲线到二维的自然推广, $n \times m$ 次贝济埃曲面的数学表达式为

$$\mathbf{r}(u, v) = \sum_{i=0}^n \sum_{j=0}^m B_{i,n}(u) B_{j,m}(v) \mathbf{r}_{ij} \quad (0 \leq u, v \leq 1) \quad (3)$$

式中 $\{B_{i,n}(u)\}$ 和 $\{B_{j,m}(v)\}$ 分别是 n 次和 m 次的伯恩斯坦基函数族。 r_{ij} ($i=0,1,\dots,n; j=0,1,\dots,m$) 是设计者事先给定的控制网格顶点。

当 $n=m=3$ 时, 曲面(3)是工程中常用的双三次贝济埃曲面, 它的矩阵表示形式是

$$r(u,v) = UMCM^T V^T \quad (0 \leq u, v \leq 1) \quad (4)$$

式中上标 T 代表矩阵的转置。

$$U = [u^3, u^2, u, 1], \quad V = [v^3, v^2, v, 1]$$

$$M = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad C = \begin{bmatrix} r_{00} & r_{01} & r_{02} & r_{03} \\ r_{10} & r_{11} & r_{12} & r_{13} \\ r_{20} & r_{21} & r_{22} & r_{23} \\ r_{30} & r_{31} & r_{32} & r_{33} \end{bmatrix}$$

贝济埃曲面对它的控制网格有逼近性质。

相应地, $n \times m$ 次有理贝济埃曲面的数学表达式为

$$r(u,v) = \frac{\sum_{i=0}^n \sum_{j=0}^m \omega_{ij} B_{i,n}(u) B_{j,m}(v) r_{ij}}{\sum_{i=0}^n \sum_{j=0}^m \omega_{ij} B_{i,n}(u) B_{j,m}(v)}$$

式中 $\{\omega_{ij}\}$ 是权因子。

B 样条曲线 1972 年至 1974 年, 美国的 Riesenfeld 等人受贝济埃曲线的启示, 把多项式 B 样条函数扩展成为参数形式的 B 样条曲线, 并使用 B 样条特征多边形来控制它的形状。工程上最常用的是三次 B 样条曲线。同贝济埃曲线相比, B 样条曲线除了有直观性和保凸性等共同的优点外, 还具有如下优点: ①更好地逼近于特征多边形, 且形状易于控制; ②局部修改只影响邻近的几段曲线, 不会牵一发动百; ③ B 样条多项式的次数较低, 计算费用小; ④ B 样条可以构造具有特殊条件的曲线, 如直线段, 拐点, 尖角等。有理 B 样条曲线同有理贝济埃曲线有着相似的数学表达式。

B 样条曲面 B 样条曲面是由 B 样条曲线通过张量积产生, 它同 B 样条曲线有着许多相似的几何性质, 有理 B 样条曲面是 B 样条曲面拓广到有理情形的曲面, 在几何外形设计中, 它是一种很有前途的曲面表示方法。

孔斯曲面 1964 年至 1967 年, 美国的 S. A. Coons 提出了用小块曲面片组合起来表示自由型曲面, 使曲面片边界上达到任意阶连续的方法, 工程中较常用的是双三次形式。在式(4)中取

$$M = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 00 & 01 & 00_v & 01_v \\ 10 & 11 & 10_r & 11_r \\ 00_u & 01_u & 00_{ur} & 01_{ur} \\ 10_u & 11_u & 10_{ur} & 11_{ur} \end{bmatrix}$$

是双三次孔斯曲面片。其中 C 称为角点信息矩阵, 它的每个元素都是矢量, 矩阵分为四块。左上角代表四个角点的位置矢量。左下角和右上角分别表示边界曲线在四个角点处沿 u 方向和 v 方向的切矢量。这三块惟一决定了四条边界曲线的形状。右下角为角点“扭矢”, 调整扭矢会引起曲面内部的隆伏, 但对曲面边界不发生影响(见图2)。

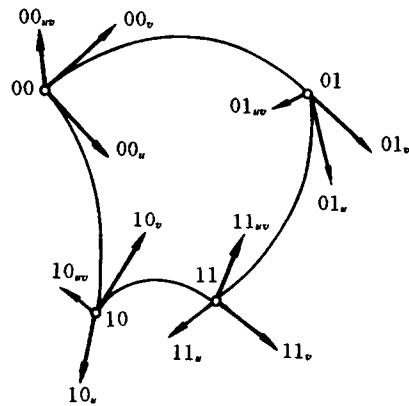


图 2

把上述小片的孔斯曲面片拼接起来, 使得连接处达到一阶或二阶偏导数连续, 便能构造各种复杂形状的几何外形, 它们在飞机外形设计中有很多应用。

双三次孔斯曲面、贝济埃曲面和 B 样条曲面都是双三次参数曲面的特殊情形。三者可通过非奇异线性变换而相互转化, 在数学上它们是等价的。但是, 从应用的角度看, 这三种曲面各有所长, 而且适用于不同的问题。

参考文献

1. 苏步青, 刘鼎元. 计算几何. 上海: 上海科技出版社, 1981
2. Bohm W. A Survey of Curve and Surface Methods in CAGD. Computer Aided Geometrical Design, 1984, 1(1): 1~60 (方遂)

jisuan lilun

计算理论 (theory of computation) 关于计算和计算机的数学理论。

1936年,为了讨论对于每个问题是否都有求解的算法,数理逻辑学家提出了几种不同的计算模型的定义。K. Gödel 和 S. C. Kleene 等人创立了递归函数论,将数论函数的算法可计算性刻画为递归可枚举性。A. M. Turing 和 E. L. Post 彼此独立地提出了理想计算机的概念,将问题的算法可解性刻画为在具有严格定义的理想计算机上的可解性。这一时期提出的通用图灵机在很大程度上影响了1946年出现的程序存储式计算机的设计思想。近代计算机的诞生使研究的焦点从理论可计算性转移到现实可计算性。因而,对一般算法设计方法的研究以及对一类问题的算法解的难度的分析与研究便成为计算机科学中的热点。由此产生了算法学和计算复杂性理论等新兴的研究领域。自 A. M. Turing 提出图灵机, E. L. Post 提出波斯特机以后,直到1948年才由美籍匈牙利数学家 J. Von Neumann 提出建立自动机的一般数学理论,对各种人造自动机和天然自动机进行比较、分析,探索其共同规律。他还研究了自动机的自繁殖和自恢复问题。1954年前后形成了时序机的概念,1963年后概率自动机的理论有了新的发展。1960年后,细胞自动机在生物学和大规模集成电路技术的基础上有了重要的进展,成为自动机理论中的一个活跃领域。形式语言的研究始于20世纪初,把形式语言用于描述自然语言则是20世纪50年代中期的事。1956年, N. Chomsky 发表了用形式语言方法研究自然语言的第一篇文章。在1960年发表的 ALGOL 60 报告中,首次使用与上下文无关文法等价的巴克斯范式系统来描述程序设计语言的局部语法,从而使形式语言理论在程序设计语言的设计与实现中发挥了重要的基础作用。

计算理论主要内容包括**算法**、**算法学**、**计算复杂性理论**、**可计算性理论**、**自动机理论**和**形式语言理论**等。

算法 解题过程的精确描述。它由有限条可完全机械执行的、有确定结果的指令(或命令、语句)构成并具有以下性质:①将算法作用于特定的输入集或问题描述上将产生由有限个动作构成的动作序列;②该动作序列具有惟一的初始动作;③除末尾的动作外,序列中的每个动作都有一个或多个后继动作;④序列或者终止于问题的解,或者终止于指出对给定输入集原问题无解。最常用的算法有**排序算法**,求解组合问题和组合最优化问题的**组合算法**等,而动作具有随机性的**概率算法**可望比确定型算法更高效。随着并行处理机的出现,一类适应于并

行操作的**并行算法**便应运而生,并且日益成为热门的研究领域。

算法学 系统地研究算法的设计、分析与验证的学科。设计是创建算法的过程,算法学的任务之一就是研究一般的有效的算法设计方法。经典的方法有:递归法、分治法、动态规划法、贪心法、回溯法和分枝限界法等。当然更多的问题还得具体情况具体分析。验证在于证明算法的正确性。基本途径是循环不变式加数学归纳法。分析是为了确定算法的效用或者说算法的复杂性以便比较求解同一问题的各种算法的优劣。

计算复杂性理论 用数学方法研究各类问题的计算复杂性的学科。算法复杂性是针对特定算法而言的,而计算复杂性则是针对特定问题而言的,后者反映的是问题的固有难度。计算复杂性等于最佳算法复杂性。最常用的**复杂性度量**有**时间复杂性**和**空间复杂性**。**复杂性归约**是比较问题的复杂性的重要工具。根据复杂性的阶可对被计算的问题分类。该学科的一个基本问题是:在计算时间多项式有界时,确定型机器与非确定型机器的解题能力是否相等?即 P 是否等于 NP?

可计算性理论 研究计算的一般性质的数学理论。它通过建立计算的数学模型,精确区分哪些问题是可以计算的,哪些问题是不可计算的。计算的过程就是执行算法的过程。根据图灵论题:图灵机可计算函数类与直观可计算函数类相同。**原始递归函数**是一类基本的直观可计算函数,但并非所有直观可计算的函数都是原始递归函数。然而,已经证明:部分递归函数类与图灵机可计算函数类相同。该学科的一个基本结论是:不可计算的函数要比可计算的函数多得多。特别地,虽然许多问题是可判定的,但更多的问题是不可判定的,如著名的停机问题和波斯特对应问题都是不可判定的。

自动机理论 以研究离散数字系统的功能和结构以及两者之关系为主要内容的数学理论。自动机是离散系统的抽象模型。给定自动机的功能描述,综合出具有这种功能的自动机的结构;给定自动机的结构,分析出它的功能,这就是自动机的综合与分析问题。自动机理论一般包括**有限自动机理论**、**无限自动机理论**、**概率自动机理论**和**细胞自动机理论**等等。

形式语言理论 用数学方法研究自然语言(如英语)和人工语言(如程序设计语言)的语法的理论。它只研究语言的组成规则,不研究语言的含义。

该学科主要研究形式语言的描述工具、文法分类、语言分类以及各类语言的性质及其间的关系等。因为自动机可看成形式语言的一种识别器,所以形式语言理论与自动机理论有着十分密切的关系。

计算理论作为计算机科学的理论基础之一,其基本思想、概念和方法广泛应用于计算机科学的各个领域。早期的通用图灵机模型就是后来的程序存储式计算机模型的基本原型。递归函数的思想用于程序设计,产生了递归过程和递归数据结构,也影响了计算机的体系结构。可计算性理论的许多结论使人们不再为求解不可判定问题而浪费时间。算法学中提出的各种一般的算法设计方法广泛应用于各种具体问题的算法设计中,算法验证技术与程序验证技术互相借鉴。算法分析技术不但有助于判断给定算法是否现实可计算,而且有助于比较算法的质量。自动机作为一种基本工具已用于程序设计语言的编译程序中。线性自动机作为伪随机序列产生器用途广泛。可逆自动机用于通信编码。细胞自动机对并行计算机的设计思想产生了明显的影响。自繁殖自动机用于研究生物发育。形式语言理论在自然语言的理解和翻译、计算机语言的描述和编译、社会和自然现象的模拟、语法制导的模式识别等方面有广泛的应用。

参考文献

1. Aho A V, Hopcroft J E and Ullman J D. The Design and Analysis of Computer Algorithms. Addison-Wesley, 1974
2. Papadimitriou C H. Computational Complexity. Addison-Wesley, 1994
3. Cutland N. Computability, An Introduction to Recursive Function Theory. Cambridge University Press, 1980
4. Hopcroft J E and Ullman J D. Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, 1979 (陈火旺 殷建平)

jisuan shulun

计算数论 (computational number theory)

研究数论中计算方法的一门学科。其中两个最核心的问题是大整数的素性检验和因子分解。现代快速计算机的出现促进了计算数论的发展,只有利用先进的计算机,很多算法才可能实现。

计算数论与现代密码学有密切的关系。一个密码系统,对合法用户来说,它的加密、解密运算应该

容易实现,但任何局外人要想破译它却是困难的。数论中就有很多计算问题,它同时具有“容易”和“困难”两个方面。例如,把两个大整数相乘是容易的,但在不知道这两个因子时,要将其乘积因子分解却是十分困难的。在 20 世纪 70 年代提出的 RSA 公钥密码正是利用了这一特性。

给定一个自然数 n ,要判断它是不是素数,这就是素性检验。最简单的办法是用不超过 \sqrt{n} 的所有素数逐个去除 n ,如果都除不尽,则 n 就是素数,否则 n 就是复合数。利用这个方法可找到 n 的因子。但当 n 很大时,这个方法的计算量太大,是不可行的。当 n 为素数时,若整数 a 与 n 互素,则

$$a^{n-1} \equiv 1 \pmod{n} \quad (1)$$

(费马小定理)。如果能够找到一个 a 使式(1)不成立,即可断定 n 是复合数。相反的结论并不能成立,即若式(1)对某个 a 成立,也不能断定 n 为素数。实际上,存在这样的复合数,使式(1)对一切与 n 互素的 a 都成立。这样的复合数称为 Carmichael 数。例如, $n = 561 = 3 \times 11 \times 17$ 就是一个 Carmichael 数。费马小定理实际上只能给出 n 为复合数的确切判断,它不能直接提供一个素性检验的有效算法。不过后来找到的很多素性检验算法,却是从费马小定理引伸发展起来的。

当式(1)成立时,称 n 为以 a 为基的伪素数。设 $n-1 = 2^k r$,其中 r 为奇数。当 n 为素数时,将式(1)两端开方,可得 $a^{(n-1)/2} \equiv \pm 1 \pmod{n}$ 。对任意 n ,若

$$a^r \equiv 1 \pmod{n}, \text{ 或存在 } i(0 \leq i < k)$$

$$\text{使 } a^{2^i r} \equiv -1 \pmod{n} \quad (2)$$

则称 n 是以 a 为基的强伪素数。若存在一个 a ,使式(2)不成立, n 就是复合数。当 n 为复合数时,在区间 $(2, n-1)$ 内至少有 $3/4$ 的数使式(2)不成立。从此区间内随机取 l 个数,对每个数检查式(2)是否成立(Rabin 检验)。若每个数都能使式(2)成立,则称 n 通过该检验。当 n 为复合数时,它通过 Rabin 检验的概率为 4^{-l} 。确切地说,这个算法实际上是概率型复合性检验。当 n 不能通过该检验时, n 就是复合数。当 n 通过该检验时,它就“很可能”是素数。

利用广义黎曼猜想可以证明,当 n 为复合数时,在区间 $(2, [2(\log n)^2])$ 内一定有一个数 a 使式(2)不成立。因此逐个检查该区间内的数能否使式(2)成立,就是一个肯定型的素性检验。但广义黎曼猜想尚未证明。

如果能将 $n-1$ 完全分解为素因子乘积,则有如下的一个肯定型素性检验,它不依赖任何未证明的数学假设。若能找到一个正整数 a ($< n$),使

$$\begin{cases} a^{n-1} \equiv 1 \pmod{n} \\ \text{对 } n-1 \text{ 的任一素因子 } p, \text{ 有 } a^{(n-1)/p} \not\equiv 1 \pmod{n} \end{cases} \quad (3)$$

则 n 是素数。这时由 a 的乘方可产生 $1, 2, \dots, n-1 \pmod{n}$, 即 $1, 2, \dots, n-1$ 都与 n 互素。仅能将 $n-1$ 部分地因子分解时,有如下的肯定型素性检验。设 s 是 $n-1$ 的因子, s 能完全地分解为素因子之积,且 $s > \sqrt{n-1}$, 若能找到一个正整数 a ($< n$),使

$$\begin{cases} a^{s-1} \equiv 1 \pmod{n} \\ \text{对 } s \text{ 的任一素因子 } p, \text{ 有 } (a^{(s-1)/p} - 1, n) = 1 \end{cases} \quad (4)$$

则 n 是素数 (Pocklington 检验)。当 $n+1$ 能完全分解时,有相应的素性检验 (Lucas-Lehmer 检验)。当 $n+1$ 仅能部分地分解到一定程度时,也有相应的素性检验。当 $n-1$ 与 $n+1$ 同时能部分分解到一定程度时,则有一种混合型的素性检验。一般地说,要将 $n-1$ 或 $n+1$ 因子分解并非易事。但当 n 具有某些特殊形状,譬如, $n = 2^{2^r} + 1$ 为费马数,或 $n = 2^p - 1$ (p 为素数) 为梅森数时,上述检验是可行的。

目前所找到的最有效的素性检验是雅可比和检验与复乘检验,这两个方法都不依赖 $n \pm 1$ 的因子分解。雅可比和检验利用代数整数环中与费马小定理类似的一个性质。复乘检验利用有限域 F_p 上的椭圆曲线。利用这两个方法,很容易检验一个几百位的整数是否是素数。

大整数因子分解的算法大体上可分为两个类型。

一个类型称为“同余式的组合”。设 n 是要分解的复合数,如果找到两个自然数 X, Y , 使 $1 \leq X, Y < n$, XY 与 n 互素,且

$$X^2 \equiv Y^2 \pmod{n} \quad (5)$$

由于 $n \mid (X-Y)(X+Y)$, 当 $n \nmid X \pm Y$ 时, $(X-Y, n)$ 就是 n 的真因子,于是计算 $(X-Y, n)$ 就可将 n 分解。大体上说,出现 $n \nmid X \pm Y$ 的概率为 $1/2$, 所以能找到形如式 (5) 的同余式越多,能分解 n 的可能性就越大。例如,若找出 10 个这样的同余式,就有 99.9% 的可能性将 n 分解。问题归结为如何构造出一批形如式 (5) 的同余式。设 $p_1 (=2), p_2, \dots, p_k$ 为最小的 k 个素数,首先设法找出一批整数 $\{a_i \mid 1 \leq i \leq l\}$, 使 $a_i^2 \equiv b_i \pmod{n}$ ($0 < b_i < n$), 其中

每个 b_i 的所有素因子都在 p_1, p_2, \dots, p_k 之中。然后在 b_1, \dots, b_l 中找出一些数 b_i , 使它们的乘积为一个平方数。把相应那些 b_i 的同余式 $a_i^2 \equiv b_i \pmod{n}$ 相乘,便得到一个形如式 (5) 的同余式。

D. H. Lehmer 和 R. E. Powers 于 1931 年提出连分数因子分解算法,利用 \sqrt{n} 的连分数展开式找到 \sqrt{n} 的一批最佳渐近分数 a_i/c_i , 设 $a_i^2 \equiv b_i \pmod{n}$, $0 < b_i < n$ 。连分数理论指出 $b_i \leq 2\sqrt{n}$, 由于 b_i 较小,就有较大的可能找出一批 b_i , 使其素因子都在最小的 k 个素数之中。称最小的 k 个素数的集合为因子基,也可以选择其他素数集合为因子基。通过因子基寻找同余式 (5) 的系统方法,是在 70 年代初由 M. Morrison 和 J. Brillhart 提出的。M. Kraitchik 于 1926 年提出利用二次函数 $f(x) = (x + [\sqrt{n}])^2 - n$ 。由于 $(x + [\sqrt{n}])^2 \equiv f(x) \pmod{n}$, $f(x)$ 相当于上述 b_i , 可以使它较小。C. Powerance 于 1982 年提出了一个系统的方法,对于相连的一串整数 x , 挑出那些素因子都在预先选定的因子基内的 $f(x)$ 。这个因子分解的算法称为二次筛法,这是目前一个有效的因子分解算法。它对于被分解的整数的形式没有特殊要求,曾用它分解 111 位的整数。它的计算量大约为 $\exp((1+o(1))(\lg n \lg \lg n)^{1/2})$ 。

大整数因子分解的另一类型算法都利用一个群,且这个群的阶 (即元素个数) 不含大素因子。设 p 是 n 的素因子, $M(k)$ 为最小的 k 个自然数的最小公倍数。若 $p-1 \mid M(k)$, 则 $2^{M(k)} \equiv 1 \pmod{p}$ (费马小定理)。如果 $2^{M(k)} \not\equiv 1 \pmod{n}$, 则 $(2^{M(k)-1}, n)$ 就是 n 的真因子,从而 n 可以分解。对于一批较小的 k , 计算 $(2^{M(k)-1}, n)$, 希望能找到 n 的一个真因子 (Pollard 算法, 1974 年提出)。这里利用了群 F_p^* (含 p 个元素的域的乘法群), 当 $|F_p^*| = p-1$ 的素因子都较小时,就有可能分解 n 。显然,也可以利用其他的群。H. W. Lenstra 于 1987 年提出椭圆曲线因子分解算法,利用了 F_p 上的椭圆曲线的点所成的群,当该群的阶适合上述要求时,就有可能将 n 分解。对于固定的一个 p , 有很多条 F_p 上的椭圆曲线可供选择使用,从而增加了分解 n 的机会。

在强化二次筛法和椭圆曲线算法的优点的基础上, John Pollard 于 1988 年提出数域筛法。1990 年分布在世界各地的很多学者通过计算机联网,在数月内,利用数域筛法共同分解了第 9 个费马数 $2^{2^9} + 1$, 它是一个 7 位素数、一个 49 位素数和一个 99 位素数的乘积,这是一个特殊形式的整数。

参考文献

1. Pomerance C, Editor. Cryptology and Computational Number Theory. American Mathematical Society, 1990
2. Koblitz N. A Course in Number Theory and Cryptography. Springer-Verlag New York Inc., 1987
3. Cohen H and Lenstra A K. Primality Testing and Jacobi Sums. Math. Comp., 1984, 48: 103 ~ 121
4. Goldwasser S and Kilian J. Almost All Primes Can Be Quickly Certified. Proc. 18th Annual ACM Symp. on Theory of Computing. 1986, 316 ~ 329
5. Lenstra H W. Factoring Integers with Elliptic Curves. Ann. of Math., 1987, (126): 649 ~ 673

(裴定一)

jisuan yuyanxue

计算语言学 (computational linguistics) 涉及语言学、计算机科学、数学和认知心理学等多学科的一门边缘学科。又称面向计算的语言研究与应用。它一方面利用计算机对语言文字进行各种定量与精密化的分析,如字频、词频统计,词类分布及其邻接关系的统计,句型研究、作品风格分析以及语法规则和语义模型的研究等;另一方面又要求研究人员为计算机处理自然语言提供计算理论或模型,以支持自然语言的分析与生成,实现自然语言人机接口、全文检索和机器翻译等应用系统。

计算语言学的研究内容大体上与“自然语言理解”相当。1990年以来,国际学术界将大规模真实文本处理作为本学科今后一个时期的战略目标。当前研究重点包括:基于复杂特征集和合一算法的语法理论、词汇语义学、大规模词库(或词汇知识库)的建设;大规模语料库(包括双语语料库)的建设;语料加工技术与工具;从经过标注的语料中获取语言知识的理论与方法;文本理解与生成等。在应用方面,机器翻译、全文检索、自动文摘、根据输入文本生成数据库记录等实用化系统已显示出巨大的市场需求。

参考文献

1. 刘开瑛,郭炳炎.自然语言处理.北京:科学出版社,1991
2. 石纯一,黄昌宁,王家庶.人工智能原理.北京:清华大学出版社,1993
3. Grishman R. Computational Linguistics. Cambridge: Cambridge University Press, 1986 (黄昌宁)

jilu leixing

记录类型 (record type) 分量可以具有不同类型的一种构造类型。记录是收集相关成分的一种数据结构。例如,关于人(person)的记录可包括名字(name)、性别(sex)及年龄(age)等,用PASCAL语言可将此记录类型定义如下:

type

```
person = record
    name: array[1..10] of char;
    sex: (male, female);
    age: 1..100
end;
```

var

```
chen, wang: person;
```

这里 name, sex, age 为域名,用来表记录中的不同分量,每个域名还有一个定义其值范围的类型。person 为定义该记录的类型的标识符,通过它可创建诸如 chen, wang 等等不同的记录实例。

记录中允许域的数据类型在表达形式上有所变化。例如,空间中点的位置可按直角坐标(x, y, z)或按极坐标(r, φ, z)的形式给定,因此在“点”类型的记录中,允许出现变化的数据形式。PASCAL 语言允许构造这种变体记录:

type

```
coordinate = (rectangular, cylindrical);
point = record
    z: real;
case c: coordinate of
```

```
rectangular: (x, y: real);
```

```
cylindrical: (r, phi: real)
```

end;

这里,变体选择符 c 描述了实际出现的可选情况。

记录类型的操作除了赋值和相等比较这两种操作外,还有记录的构成与选择,即从各分量的值来构造出的记录类型的一个值,和从记录类型的一个值中选择出其组成成分。(陈涵生)

jicheng

继承 (inheritance) 基于层次关系的不同类间共享数据和操作的关系。通过继承,一个类的定义可以基于另一个类(后者称作前者的基类):对基类

的特征或是不加任何修改,或是根据特定情形作适当调整。

作用 继承是面向对象软件开发的重要设施,它有效地支持软件复用与扩充。类既是模块,又是类型。对应于类的这两种作用,继承可作为:①模块扩充机制,它通过增加或修改已有类的特征来定义新类;②类型精化机制,它支持通过例化已有类型来定义新类型。

种类 根据继承关系的性质,可区分如下几种继承:

(1) 直接继承和间接继承:如果类 C 的定义直接使用类 B 中的特征,则称 C 直接继承 B,且称 B 为 C 的直接基类, C 为 B 的直接衍类。如果类 C 直接继承类 B,类 B 直接继承类 A,则称 C 间接继承 A,且称 C 为 A 的间接衍类, A 为 C 的间接基类。间接继承体现了继承关系的可传递性。

(2) 单继承和多继承:如果一个类只有一个直接基类,则称该继承关系为单继承;如果一个类有多个直接基类,则称该继承关系为多继承。

(3) 重复继承:若类 D 的多个直接基类有共同基类 A 时,则称 D 重复继承 A,称 A 是 D 的重复

基类。

参考文献

1. 徐家福等. 对象式程序设计语言. 南京: 南京大学出版社, 1992
2. Special Issue on Object-Oriented Design. CACM, 1990, 33(9) (张家重 王志坚)

jjiafaqi

加法器 (adder) 对以数字形式表示的两个 n 位数字求和的一种运算电路。加法器是计算机中最基本的运算部件。

基本的加法单元称为全加器,它要求三个输入量,即操作数 X_n 、 Y_n 和低位传来的进位 C_{n-1} ,并产生两个输出量,即本位和 F_n 及向高位的进位 C_n , F_n 和 C_n 的表达式为

$$F_n = X_n \bar{Y}_n \bar{C}_{n-1} + \bar{X}_n Y_n \bar{C}_{n-1} + \bar{X}_n \bar{Y}_n C_{n-1} + X_n Y_n C_{n-1}$$

$$C_n = X_n Y_n \bar{C}_{n-1} + X_n \bar{Y}_n C_{n-1} + \bar{X}_n Y_n C_{n-1} + X_n Y_n C_{n-1}$$

全加器的功能表见图 1(a), 逻辑图见图 1(b), 逻辑符号图见图 1(c)。

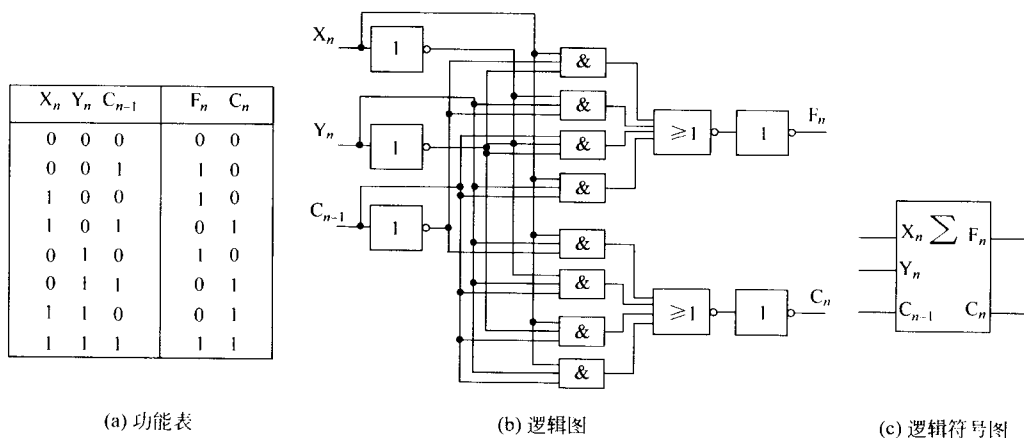


图 1 全加器的功能表、逻辑图及逻辑符号图

n 位加法器可分为串行加法器和并行加法器。在串行加法器中,只用一位全加器,数据逐位串行进入加法器进行运算。串行加法器具有器件少、成本低的优点,但运算速度太慢,在计算机中几乎不使用。并行加法器的位数和操作位数相同,能够在一步操作中对各位同时相加,影响加法时间的主要因素是进位信号的传递时间。

按进位链来分,并行加法器可分为串行进位和

并行进位两种。

并行加法器的每一位全加器都有一个从较低位送来的进位和一个向较高位的进位,将各位之间的进位线路连接起来,构成进位链。每一位的进位表达式为

$$C_i = A_i \cdot B_i + (A_i \oplus B_i) \cdot C_{i-1}$$

其中: $A_i \cdot B_i$ 取决于本位参加运算的两个数,而与低位进位无关,因此称 $A_i \cdot B_i$ 为进位产生函数(本位进位产生),用 G_i 表示; $(A_i \oplus B_i) \cdot C_{i-1}$ 则不但与本位的

两个数有关,还依赖于低位送来的进位,因此称 $A_i \oplus B_i$ 为进位传递函数(低位进位传递),用 P_i 表示。

将 n 个全加器相连可得 n 位串行进位加法器(图2),其加法时间较长。这是因为其位间进位是

串行传递的(也称之为行波进位),本位全加和 F_i 的操作必须等低位进位 C_{i-1} 到来后才能进行,加法时间与位数有关。并行进位又叫先行进位或超前进位,特点是各级进位信号同时形成,其值如下:

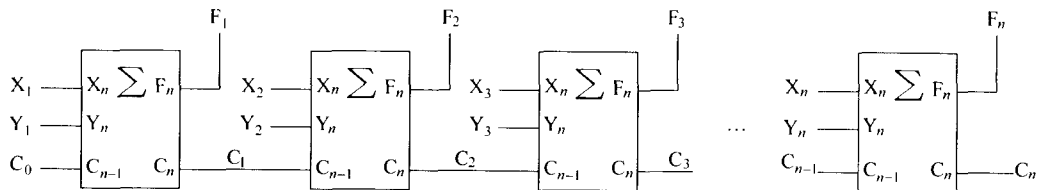


图2 串行进位加法器

$$C_1 = G_1 + P_1 \cdot C_0$$

$$C_2 = G_2 + P_2 \cdot C_1 = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot C_0$$

$$C_3 = G_3 + P_3 \cdot C_2 = G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot G_1 + P_3 \cdot P_2 \cdot P_1 \cdot C_0$$

$$C_4 = G_4 + P_4 \cdot C_3 = G_4 + P_4 \cdot G_3 + P_4 \cdot P_3 \cdot G_2 + P_4 \cdot P_3 \cdot P_2 \cdot G_1 + P_4 \cdot P_3 \cdot P_2 \cdot P_1 \cdot C_0$$

⋮

所有各位的进位都直接从 C_0 并行产生,这种进位方式速度快,但是随着位数的增加, C_i 的逻辑表达式变得越来越长,这会使电路结构变得很复杂。通常采用分组并行进位方式,它是把 n 位字长分为若干小组,在组内各位之间实行并行快速进位,而在组间可以采用串行进位方式,或是采用并行快速进位方式。

美国 Intel 公司的 74283 芯片就是一个 4 位超前进位加法器,利用 4 片 74283 就可组成一个 4 位一组、组内并行进位、组间串行进位的 16 位加法器。

参考文献

1. 蒋本珊. 电子计算机组成原理. 北京: 北京理工大学出版社, 1994
2. 王爱英主编. 计算机组成与结构(第三版). 北京: 清华大学出版社, 2001 (刘恩德)

jiagu jishu

加固技术 (ruggedization technology) 为使产品适应恶劣环境或确保其使用安全而采取的防护技术措施。加固技术主要包括抗振动、抗冲击技术、热设计技术、电磁兼容设计技术、抗核爆炸技术及三防(防潮湿、防沙尘、防烟雾和霉菌)技术和防信息泄漏技术等。

常用的加固方法有先天加固与后天加固。

先天加固又称为内加固,指按照加固要求进行

元器件选择、电源、电路板及机箱设计,从而使加固后的产品能满足恶劣环境要求。先天加固的优点是加固性能好,可适用于最恶劣的环境。缺点是价格昂贵,技术、工艺复杂,对电路及元器件的要求高,研制周期长。

后天加固又称为外加固,是利用优选的民用产品,采用重新设计机箱以及某些能达到高温、低温、潮湿、冲击、振动要求的措施,从而满足初级加固环境要求。后天加固的优点是成本低,研制周期短,缺点是不能用于较为恶劣的环境。

根据加固程度和使用环境的不同,加固型产品可分为军用规范型(简称 M 型)、加固型(简称 R 型)、防信息泄漏型(简称 T 型)三种类型。每一类型又可分为几个级别,如地面固定、车载、舰载、有(无)空调、野外等。

军用规范型是严格按照一系列军用标准设计和制造的,可满足最恶劣的外部环境要求。

加固型一般是采用后天加固的方法,使之满足一般恶劣环境的要求。

防信息泄漏型产品是指满足计算机信息泄漏要求的专用型产品。防信息泄漏型产品不一定是军用规范型或加固型产品。(於亮)

jiashhe jianyan

假设检验 (hypothesis testing) 构造一个统计量,根据给定的子样计算它的值,从而判断有关母体分布假设是否正确的统计推断方法。当母体分布函数形式未知时,如要检验母体分布是否为某种形式的分布,则称为非参数假设检验;当母体分布形式已知,但其中有未知参数,如检验这些参数是否满足某种假设,则称为参数假设检验。下面主要叙述参数假设检验。设母体分布为 $F(x; \theta)$, θ 为未知参数,

其取值空间为 Θ 。首先要提出原假设 $H_0: \theta \in \Theta_0$; 对立假设 $H_1: \theta \in \Theta_1$ (Θ_0, Θ_1 均为 Θ 的子集)。假设检验问题就是要构造一个统计量 $T = T(X_1, \dots, X_n)$, 根据子样的取值 (x_1, \dots, x_n) , 计算 $T(x_1, \dots, x_n)$, 并根据它的值来确定子样空间 \mathcal{X} 的一个划分; $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1, \mathcal{X}_0 \cap \mathcal{X}_1 = \emptyset$, 其中 \mathcal{X}_0 称为否定域, \mathcal{X}_1 称为接受域。当 $(x_1, \dots, x_n) \in \mathcal{X}_0$ 时否定 H_0 , 否则接受 H_0 。这里划分的根据是: 当 $\theta \in \Theta_0$ 时, $p_\theta(\mathcal{X}_0) \triangleq p_\theta((X_1, \dots, X_n) \in \mathcal{X}_0)$ 为一个很小的数 α , 通常取 $\alpha = 0.05, 0.01$ 等。根据小概率事件原理: “小概率事件在一次试验中是几乎不可能发生的”, 既然 $p_\theta(\mathcal{X}_0) = \alpha$, 而今居然观察到 $(x_1, \dots, x_n) \in \mathcal{X}_0$, 所以有理由认为 θ 不属于 Θ_0 , 也即否定 H_0 。但这种判断有可能错误, 称 $\alpha = p_\theta(\mathcal{X}_0)$ 为犯第一类错误 (弃真) 的概率。同时, 当 $\theta \in \Theta_1$, 即 H_0 不真时, 也希望判断错误的概率 $p_\theta(\mathcal{X}_1)$ 尽可能地小, 称它为犯第二类错误 (存伪) 的概率, 记为 β 。在样本容量 n 固定时, 不可能使 α 与 β 同时任意地小, 一般只能在满足对 α 要求的条件下, 寻求使势函数 $p_\theta(\mathcal{X}_0) = 1 - \beta(\theta \in \Theta_1)$ 尽可能地大。特别要指出的是, 因为 α 很小, 所以当原假设 H_0 为真时, 否定 H_0 的概率很小, 实际上起着保护原假设的作用, 因此在应用中, 总是把按过去的经验很可能为真的或者特别要注意排除的, 取为原假设。称 α 达到要求的, 势函数达到最大的检验为最佳检验。奈曼-皮尔逊引理指出: 设母体分布密度为 $f(x; \theta)$, θ 未知, 检验问题 $H_0: \theta = \theta_0, H_1: \theta = \theta_1$, 则对任一常数 $\alpha (0 < \alpha < 1)$, 存在显著水平 (即犯第一类错误概率) 为 α 的最佳检验法, 它的拒绝区域 \mathcal{X}_0 , 由下式确定

$$l(x_1, x_2, \dots, x_n) = \frac{\prod_{i=1}^n f(x_i; \theta_1)}{\prod_{i=1}^n f(x_i; \theta_0)} \geq k \quad (1)$$

其中 k 为满足 $p_{\theta_0}(l(x_1, x_2, \dots, x_n) > k) = \alpha$ 的数。在奈曼-皮尔逊引理中, Θ_0, Θ_1 均为单点集, 这种检验称为简单假设检验, 否则称为复合假设检验。当 H_1 为复合假设时, 不一定存在最佳检验法。可以证明, 当 H_1 是单边的情形, 比如 $\theta < \theta_1$ 或 $\theta > \theta_1$, 则存在一致最佳检验 (即对一切 $\theta > \theta_1$, 势函数 $p_\theta(\mathcal{X}_0)$ 都达到最大)。

最常用的参数检验法是 u 检验法、 t 检验法与 χ^2 检验法。 u, t 检验法都是用来检验正态母体中关于均值的假设, 前者适用于方差已知的情形, 后者用于方差未知的情形。 χ^2 检验则用来检验正态母体

关于方差的假设。

(1) 设有正态母体 $N(\mu, \sigma_0^2)$, σ_0^2 已知, 检验 $H_0: \mu = \mu_0, H_1: \mu \neq \mu_0$ 。可以证明当 H_0 成立时, 统计量 $(\bar{X} - \mu_0) / \frac{\sigma}{\sqrt{n}}$ 服从 $N(0, 1)$ 分布, 因此可得拒绝区域为: $|\bar{X} - \mu_0| > N_{\alpha/2} \sigma / \sqrt{n}$, 其中 $N_{\alpha/2}$ 为正态分布的 α 临界值, 即正态 $N(0, 1)$ 变量 X 大于 $N_{\alpha/2}$ 的概率为 $\alpha/2$ 。由此可得关于 μ 的置信度为 $1 - \alpha$ 的置信区间为

$$(\bar{X} - N_{\alpha/2} \sigma / \sqrt{n}, \bar{X} + N_{\alpha/2} \sigma / \sqrt{n}) \quad (2)$$

(2) 上述问题中, 若 σ^2 未知, 则可证明当 H_0 成立时, 统计量 $T = \sqrt{n}(\bar{X} - \mu_0) / s_0$ 服从 t_{n-1} 分布, 其中 s_0^2 为子样的修正方差 $s_0^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2$ 。于是可得拒绝域为: $|\bar{X} - \mu_0| > (s_0 / \sqrt{n}) t_{n-1}(\alpha/2)$, 其中 $t_{n-1}(\alpha)$ 为 t_{n-1} 分布的临界值, 可从 t 分布表查得。由此可得关于 μ 的置信度为 $1 - \alpha$ 的置信区间是

$$(\bar{X} - t_{n-1}(\alpha/2) s_0 / \sqrt{n}, \bar{X} + t_{n-1}(\alpha/2) s_0 / \sqrt{n}) \quad (3)$$

(3) 设正态母体 $N(\mu, \sigma^2)$ 中 μ 已知, 检验 $H_0: \sigma^2 = \sigma_0^2, H_1: \sigma^2 \neq \sigma_0^2$ 。在 H_0 为真时, $k = (1/\sigma_0^2) \sum_{i=1}^n (X_i - \mu)^2$ 服从 χ_n^2 分布, 因此可得拒绝区域为 $k > \chi_n^2(\alpha/2)$ 或 $k < \chi_n^2(1 - \alpha/2)$, 其中 $\chi_n^2(\alpha)$ 为 χ_n^2 分布的临界值, 可从 χ^2 分布表查得。 σ^2 的置信度为 $1 - \alpha$ 的置信区间是

$$\left(\sum_{i=1}^n (X_i - \mu)^2 / \chi_n^2(\alpha/2), \sum_{i=1}^n (X_i - \mu)^2 / \chi_n^2(1 - \alpha/2) \right)$$

(4) 上述问题中若 μ 未知, 则当 H_0 为真时, $ns^2 / \sigma_0^2 \sim \chi_{n-1}^2$, 拒绝域为 $ns^2 / \sigma_0^2 > \chi_{n-1}^2(\alpha/2)$ 或 $ns^2 / \sigma_0^2 < \chi_{n-1}^2(1 - \alpha/2)$ 。 σ^2 的置信度为 $1 - \alpha$ 的置信区间为

$$(ns^2 / \chi_{n-1}^2(\alpha/2), ns^2 / \chi_{n-1}^2(1 - \alpha/2)) \quad (4)$$

比较两个正态母体方差的大小还有 F 检验法。对于二元正态母体, 可以用 t 检验法检验其相关系数是否为 0, 也可用 χ^2 检验法检验两个分量是否独立。最常用的分布检验法是皮尔逊 χ^2 检验法, 柯尔莫哥洛夫检验法, 斯米尔诺夫检验法。

参考文献

1. 陈希孺. 数理统计引论. 北京: 科学出版社, 1981
2. 中山大学数力系. 概率论与数理统计 (下册). 北京: 人民教育出版社, 1980 (金治明)

jia tuoji

假脱机 (simultaneous peripheral operations online, spool) 使独占使用的设备变成可共享设备的虚拟设备技术。

在 20 世纪 50 年代后期, 计算机系统中的一个主要问题是快速的中央处理器和慢速的外围设备的工作速度不匹配, 为了解决这一问题引入了脱机输入输出技术。在这种系统中, 计算机的输入输出操作都通过磁带进行。同时, 另外使用一台小机器将用户准备好的程序和数据等输入到磁带上, 然后将这盘磁带安装到计算机的磁带机上作为其输入带, 而计算机又将处理结果送到输出带上。最后, 那台小机器再将输出带上的信息送到某种输出设备(例如, 行式打印机)上。这样, 脱机输入输出技术用快速磁带机代替了低速设备(例如卡片输入机、行式打印机等), 从而缓和了高速中央处理器和低速输入输出设备之间的矛盾。随着计算机处理能力的增加以及多道程序、通道技术的引入, 原先用一台小机器脱开计算机进行的输入输出操作, 现在一般也由计算机完成, 故称为联机的同时外围设备操作, 又称为假脱机输入输出, 为实现此功能所配置的软件系统则称为假脱机系统。

假脱机输入输出是利用快速、大容量、可为多个进程共享的磁盘设备作为中介, 模拟多个非共享的字符输入输出设备。于是, 需要独占使用这种字符设备的进程不再需要直接使用物理设备, 而代之以这种虚拟设备进行信息交换, 虚拟设备和物理设备之间则在系统特设的假脱机进程控制下进行信息传输。

以行式打印机为例, 进程打开行式打印机时, 系统在磁盘上为其创建一个虚行式打印文件, 该进程对行式打印机的输出都写到这个文件上。进程释放行式打印机时, 系统就将相应的虚行式打印文件送到同类文件的队列中, 而行式打印假脱机进程则不断从该队列中取出文件并将它们送到行式打印机上打印。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989

(徐良贤)

jiancai guocheng

剪裁过程 (tailoring process) 对软件过程和活动实施剪裁的过程。将一选定模型以及相关标准

应用于某一领域或具体软件项目, 形成该领域的模型及标准或该软件项目的各个软件过程和活动。最初选定的模型是相对抽象、具有相对普遍性的, 而所选标准是描述活动全集的, 将它们针对某领域剪裁意味着形成该领域的相对特殊的模型及标准; 将它们针对软件项目进行剪裁意味着形成该项目的软件过程和活动, 这是剪裁过程的双重意义。

剪裁过程的基本内容包括剪裁过程的各项活动和剪裁的指导原则。剪裁过程的各项活动:

(1) 指明项目环境 指明影响剪裁的项目环境特征。如使用的模型和方法; 系统和软件需求; 机构的政策、步骤和策略; 系统和软件的大小、重要性和类型; 参加人员和合作伙伴的数量等。

(2) 收集信息 应向所有会影响剪裁的机构收集信息。用户、支持人员、负责签订合同的人员、潜在的投标者应参与剪裁。

(3) 选取过程、活动和任务 根据上述收集到的数据, 决定要实施的过程、活动和任务。

(4) 将剪裁的决定及其理由编成文档 所有剪裁决定及这些决定的理由应编成文档。

剪裁的指导原则: 剪裁可分为二级, 第一级是根据应用领域的不同进行剪裁, 第二级是根据每一个具体的项目或合同进行剪裁。这里给出实施第一级剪裁的一些主要原则。

(1) 开发过程的剪裁 ①对嵌入或集成在系统中的软件, 应考虑该过程中的所有活动, 并且必须明确开发者是否要实施或支持系统的活动; ②对于独立的软件, 关于系统的活动可能不需要, 但应加以考虑。

(2) 和评价有关的活动的剪裁 评价可以分成以下 5 类。前面 4 类评价属项目级, 最后 1 类属机构级。应根据项目或机构的范围、大小、复杂性和重要性相应地选取和剪裁这些评价: ①过程内部的评价: 由实行过程内指定任务的人员进行的评价; ②验证和确认: 由获取者、供应者或一个独立的合作伙伴根据项目从不同的深度验证和确认软件产品; ③联合评审和审计: 由评审方和被评审方联合评价软件产品和活动的状况; ④质量保证: 由和软件开发没有直接关系的人员进行, 目的是独立地保证软件产品符合合同要求并按制订的计划执行; ⑤改进: 由一个为了有效管理和自我改进其过程的机构进行, 与项目或合同的要求无关。

(3) 剪裁和应用软件过程的若干考虑 以下是剪裁和应用软件过程时需指明、考虑的一些关键的

项目特征: ①机构政策: 指明机构的政策, 如使用的计算机语言、安全性和保密性、硬件储备要求、风险管理等。②获取策略: 为项目指明获取策略, 如合同类型、合同方和验证、确认代理的参与、获取者参与合同的程度、合同者能力的评价等。③支持的概念: 指明支持的概念, 如期望的支持程度、更改程度、获取者或供应者是否提供支持等。④生存周期模型: 指明项目使用的生存周期模型, 如瀑布模型、迭代瀑布模型、演化模型、螺旋模型等。⑤合作伙伴: 指明参与项目的合作伙伴以及人员的数目。对一个有许多合作伙伴和数十或数百人参加的大型项目, 需要加强管理和控制。内部和独立的评价、评审、审计等是大型项目的重要工具, 而对小型项目, 这些控制可以大大减少。⑥系统级特征: 指明系统级特征, 如子系统数目和配置项等。若系统有许多子系统或配置项, 则开发过程应对每个子系统或配置项仔细进行剪裁, 并应考虑所有的界面和集成方面的要求。⑦软件级特征: 指明软件级特征, 如软件配置项的数目、类型、大小以及软件的重要性、技术风险等。若软件有许多软件配置项、部件和单元, 则开发过程应对每个软件配置项仔细进行剪裁, 并应考虑所有的界面和集成方面的要求。⑧风险性: 软件开发可以有技术风险。如果所使用的软件技术不成熟, 所开发的软件太复杂或软件包含安全性、保密性或其他关键性要求, 则需要有严格的规约、设计、测试和评价。这时, 独立的验证和确认将显得更加重要。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes — IEEE Std. 1074 ~ 1991
2. ISO / IEC 12207: 1995 Information Technology — Software — Part 1: Software Life-Cycle Process

(黄嘉启)

jiandan leixing

简单类型 (primitive type) 其值不能进一步分解为更简单的值的类型。又称基本类型或初等类型。

程序设计语言中简单类型的选择与该语言预期的应用领域有关, 一般包括实型、整型、字符型和布尔型, 它们往往是语言预定义的类型。实型值是实现定义的有理实数的一个子集, 整型值是实现定义的整数的一个子集, 字符型值是实现定义的一个字符集, 布尔型的值可由字面常量或预定义的常量标识符 false 和 true 来定义。有些语言还允许用户自

行定义简单类型, 如在 PASCAL 和 Ada 中, 用户可自定义枚举类型和子域类型, 枚举类型是由通过列举有限多个标识符来定义的, 这里每一个标识符 (枚举符号) 即表示一个枚举值, 子域类型是通过指定一个已有类型的子集来定义的。例如:

```
Type color = ( red, green, blue );
sub = 28. . . 31
```

离散简单类型是其值与整型 (或其一个子域) 有一一对应关系的一种简单类型。在 PASCAL 和 Ada 中, 离散简单类型的值可以用于诸如计数、分支选择和数组下标中, 而在其他语言中, 仅整型可用于此类操作。

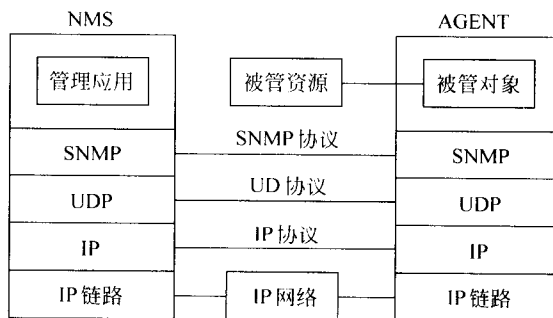
参考文献

Watt D A. Programming Language Concepts and Paradigms. Prentice Hall, 1990 (金凌紫 陈涵生)

jiandan wangluo guanli xieyi

简单网络管理协议 (Simple Network Management Protocol, SNMP) 基于 TCP/IP 协议的一种功能比较简单的网络管理协议 (参见 TCP/IP 协议集)。

简单网络管理协议 (SNMP) 的体系结构由 3 部分组成, 即 SNMP 协议本身、管理信息结构 (SMI) 和管理信息库 (MIB)。它们运行于 TCP/IP 协议栈上, 是应用层的一部分。SNMP 中对网络管理资源的描述采用面向对象的方法。图 1 示出了 SNMP 的体系结构。



NMS —— 网络管理站
AGENT —— 代理
SNMP —— 简单网络管理协议
UDP —— 用户数据报协议
IP —— 网际协议

图 1 SNMP 体系结构

管理应用和被管对象是 SNMP 协议的用户。被管对象是被管资源在网络管理中的表现, 它体现了

被管资源的特性。被管资源的变化应该在相应的被管对象中得到体现,反之,被管对象的变化也能引起被管资源的相应变化。

SNMP 协议向管理应用或被管对象提供统一的管理操作,包括 GET(取)、SET(设置)和 EVENT(事件)。SNMP 协议的低层协议是用户数据报协议(UDP)。

管理信息结构(SMI)给出了一组规则,用于如何定义被管对象、如何访问被管对象以及如何把这些对象添加到管理信息库(MIB)中去。

SNMP 中定义的部分管理信息库(MIB)中的被管对象组如表 1 所示。

表 1 被管对象组

被管对象组	含 义
system	被管理的系统本身信息
interface	本结点上的接口信息
at	ip 地址映射信息
ip	ip 协议信息
icmp	icmp 协议信息
tcp	tcp 协议信息
udp	udp 协议信息
egp	egp 协议信息
其他	其他信息

注: ip——网际协议(IP); icmp——因特网控制消息协议(ICMP); tcp——传输控制协议(TCP); udp——用户数据报协议(UDP); egp——外部网关协议(EGP)。

已定义的被管对象组达一百多个,其中部分是必须实现的(如表 1 中前 5 项)。

SNMP 系统是实现 SNMP 的网络管理系统。它由网络管理站(NMS)和代理(agent)组成。SNMP 系统支持 3 种基本类型命令: GET、SET 和 EVENT。网络管理站(NMS)通过 GET 查询代理(agent)中管理信息库(MIB)内的信息,它还可以通过 SET 要求 agent 执行指定的工作。同时, agent 也能通过 EVENT 主动把网络的某些信息(如报警信息等)传送给网络管理站(NMS)。NMS 和 agent 之间通过执行上述 3 种命令构成的操作来交换信息, NMS 向操作员提供多种多样的界面,不受 SNMP 约束。

参考文献

Stallings W. SNMP, SNMPv2, SNMPv3, and RMON 1 and 2. Third Edition. Addison - Wesley, 1999
(钱松荣)

jianzhuqun zixitong

建筑群子系统(campus subsystem) 结构化布线系统中由连接楼群之间的通信传输介质及各种支持设备组成的子系统。建筑群子系统也称为户外子系统。其传输介质除了各种有线手段之外,还可包含其他无线通信手段,如微波、无线电通信等。

户外电缆在进入大楼时通常在入口处经过一次转接接入户内系统,如图 1 所示。在转接处可以加上电器保护设备。现代化电话通信系统中通信线路在进入楼群时一般都考虑这一点,主要是避免因雷击或与高压线接触而给人和设备安全带来的损失。

建筑群子系统布线方式有以下几种:地下管道敷设方式、直埋沟内敷设方式、架空方式等,不同方式的优缺点如下表所示:

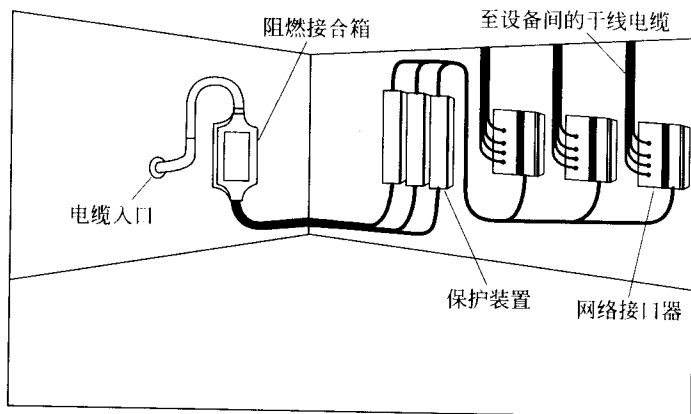


图 1 建筑物缆线入口区

方 式	优 点	缺 点
管道内	提供最佳的机构保护,任何时候都可以敷设电缆,电缆的敷设和扩充都很容易,能保持道路和建筑物的外貌整齐	挖沟、开管道、建入孔的初次投资较高
直埋	提供某种程度的机械保护,保持道路和建筑物的外貌整齐,初次投资较低	扩容和更换电缆时会破坏道路和建筑物的外貌整齐
架空	如果本来有电杆,则成本最低	没有提供机械保护,安全性差,影响建筑物美观

参考文献

胡道元主编. 网络设计师教程. 北京: 清华大学出版社, 2001
(王晓东)

jianbie

鉴别 (authentication) 验证某个通信参与方的身份是否与他所声称的身份一致的过程。一般通过某种复杂的身份认证协议来实现。身份认证协议是一种特殊的通信协议。它定义了参与认证服务的所有通信方在身份认证过程中需要交换的所有消息的格式、发生的次序以及消息的语义。通常采用密码学机制,例如用加密算法来保证消息的完整性、保密性。身份认证是建立安全通信的前提条件,只有通信双方相互确认对方身份后才能通过加密等手段建立安全信道,同时它也是授权访问(基于身份的访问控制)和审计记录等服务的基础,因此身份认证在网络安全中占据十分重要的位置。身份认证协议在解决分布式,尤其是开放环境下的通信安全问题起着很重要的作用。

口令技术是常用的一种身份认证手段。使用口令存在的最大问题是口令的泄露。口令泄露可以有多种途径,例如,登录时被他人看见;攻击者可能从计算机中存放口令的文件中读到;可能被在线攻击猜测出;也可能被离线攻击搜索到。所谓在线攻击是指在线状态下攻击者对用户口令进行的猜测攻击。所谓离线攻击是指攻击者在离线状态下通过某些手段进行穷极式的口令猜测,采用攻击字典和攻击程序,以最终获得口令。这种离线攻击方法是 Internet 上常见的攻击手段。

另一种身份认证技术是采用物理形式的身份认证标记来进行身份认证。常用的身份认证标记是磁卡和智能卡。磁卡储存着关于用户身份的一些数据,用户只有通过读卡设备向联网的认证服务器提供口令才能证明自己的身份。最简单的智能卡称作个人认证号(PIN)保护记忆卡。PIN 是由数字组成的口令,只有读卡机将个人认证号输入保护记忆卡

后才能读出卡中保存的数据。这种卡比磁卡安全,可以存放一些秘密信息。一种智能卡,称作加密挑战-响应卡。卡中有一个加密密钥,可使用该密钥进行加密和解密,但该密钥是无法被读出的。这种智能卡通常采用公钥算法(参见公钥密码技术),储存的是用户的私钥,可在离线状态下进行认证。在与计算机进行交互时首先递交代表自己身份的公钥证书,计算机验证证书的签发者后就获得了用户的公钥。

基于密码学原理的密码身份认证协议比基于口令或者地址的认证更加安全,而且能够提供更多的安全服务。

身份认证协议一般有两个通信方,可能还会有一个双方都信任的第三方。其中一个通信方按照协议的规定向另一方或者第三方发出认证请求,对方按照协议的规定作出响应或者其他规定的动作。当协议顺利执行完毕时,双方应该确信对方的身份。

在很多协议中,不仅要求验证相互身份,而且还要建立后续通信使用的会话密钥。所谓会话密钥是指在一次会话过程中使用的密钥,一般都是由机器随机生成的,但在实际使用时会话密钥往往是在一定时间内都有效,并不真正限制在一次会话过程中。虽然公钥算法也被用于认证协议中,但是由于公钥算法复杂度高,大量数据的加密还是采用传统密码,因此会话密钥都是传统密钥。因为会话密钥主要用于通信加密,因此也被称为通信密钥,以便与用于身份认证的认证密钥加以区分。会话密钥只在一定时间内使用并且是机器随机生成的,能够有效地抵抗密码分析攻击。认证密钥必须尽量减少使用,否则容易被攻击者搜集到足够的密文数据进行密码分析。需要建立会话密钥的认证协议也被称为密钥分发协议。

根据使用的认证密钥类型,常见的身份认证协议可以分为基于对称密钥和基于公钥两大类。基于对称密钥的安全协议需要在通信方之间事先建立好共享的密钥,也称共享密钥方式的安全协议;基于公

钥的安全协议则事先需要知道对方的公钥。

(1) 共享密钥认证

共享密钥认证思想是从口令认证用户发展来的。传统加密体制是检验对方传递来的口令是否正确,但是,口令容易被窃听而泄露。必须采用既能够验证对方拥有共同的秘密又不会在通信中泄露该秘密的方法,挑战-响应技术被用来达到这一目的。

在网络环境下一台计算机可能要与很多台计算机进行身份认证,如果全部采用挑战-响应方式认证,那么就需要与众多的计算机都建立共享密钥。这样做在大型网络环境中既不经济也不安全,同时大量共享密钥的建立、维护和更新将是非常令人头疼的事情。

Needham 和 Schroeder 在 1978 年提出**密钥分发中心(KDC)**的概念。密钥分发中心(KDC)在网络环境中为大家所信任,并且与每个网络通信方都有一个共享密钥。网络中每个通信参与方之间的认证需要借助于可信的第三方 KDC 才能完成,它们都只与 KDC 有共享密钥。KDC 负责通信双方创建并分发共享密钥,通信双方获得共享密钥后再利用挑战-响应协议建立相互信任关系。

(2) 公钥算法

公钥算法的出现为身份认证协议带来了更强有力的方法和手段,因为可以让对方通过密码运算验证自己的身份而不需要将自己的私钥告诉对方。在公钥算法中,一般将利用私钥对明文信息进行的变换称为**签名**,变换后的信息为签名信息。利用公钥对明文信息的变换称为**封装**或者加密。

使用公钥方式进行身份认证时需要事先知道对方的公钥。虽然已经有算法可以解决双方在通信时直接交换公钥的安全问题,但是从使用和管理方便程度出发,则需要一个依靠可信第三方来参与分发公钥。如果没有可信第三方的参与,每个通信参与方都需要记住所有其他用户的公钥,这不仅增加负担而且无法更新维护;同时由于每个通信方产生自己的私钥和公钥,而它们的可信赖程度不同,一旦出现问题 and 纠纷则需要权威中间机构进行仲裁。在实际网络环境中采用证书的方式来分发公钥。证书是一种特殊格式的数据记录。它包含有证书代表的通信参与方的名字、身份信息、公钥以及签发机构、签发日期、序列号、有效期等相关数据,由**证书权威机构(CA)**用自己的私钥进行签名。证书权威机构扮演可信第三方的角色。它是大家信任的组织或机构。所有的公钥认证系统都采用了证书方式。证书

被设计存放在目录服务系统中,通信参与方拥有证书权威机构的公钥,可以从目录服务中获得通信对方的证书,通过验证证书权威机构签名可以相信证书中列出的对方公钥。

采用密钥分发中心(KDC)方式和证书权威机构(CA)方式是分发密钥的主要技术。它们各有自己明显的优势和缺陷。公钥方式的身份认证协议安全强度高,但是计算机开销大,因此越来越多的安全系统倾向于利用公钥进行认证和建立对称的会话密钥,利用传统密钥进行大量数据传输的方法,例如 SSL 协议、PGP 等。

参考文献

Needham R and Schroeder M. Using Encryption for Authentication in Large Networks of Computers. Communications of the ACM, 1978, 21(12) (胡道元)

jianma

键码(key) 实体的一个属性或一组属性,其值可用来惟一标识该实体。又称**键**。

在数据库领域中,当用概念模型描述现实世界客观事物时,将客观存在并可互相区分的事物称作**实体**,实体所具有的特性称作**属性**,能够惟一标识实体的一个或一组属性称作**键码**。例如学生实体可有学号、姓名、性别、所在系、年级等属性,其中学号是键码。

在关系数据库系统中,基于**数据依赖**概念可以更严格地定义键码如下:设有关系模式 $R\langle U, F \rangle$,其中 U 为属性集合, F 为 U 上的函数依赖集合。若有属性组 $K \subseteq U$,满足 $K \rightarrow U \in F^+$,而 K 的任何真子集 K' 都不能满足 $K' \rightarrow U \in F^+$,则称 K 为 R 的**候选键码**。当候选键码多于一个时,则选定其中一个为主键码。

在关系数据库系统中,如果关系 R_2 的一个属性(或属性组)的值与另一个关系 R_1 的主键码的值相对应,则称 R_2 中的这个属性(或属性组)为**外键码**。(说明, R_1 和 R_2 不一定是两个不同的关系)。例如,下面的两个关系

部门(部门号,部门名,部门地址)

职工(职工号,职工名,部门号)

“职工”关系的“部门号”属性与“部门”关系的主键码“部门号”相对应,因此“职工”关系中的“部门号”为外键码。

键码是数据库系统中的重要概念。它是信息检索的依据。给定键码值作为检索条件,则检索的结

果是至多一个记录或元组值。

关系数据库中外键码与主键码的对应是关系之间联系的体现,它反映了一个关系中的元组对另一个关系中的元组的参照。

参考文献

1. Date C J. An Introduction to Database System. 4th Ed. Addison-Wesley, 1986
2. Ullman J D. Principles of Database and Knowledge - Base Systems. Vol. 1. Computer Science Press, 1988
(唐世渭 杨冬青)

jianpan

键盘 (keyboard) 由一组具有某些功能的键组合而成的输入设备。使用者通过击键向计算机输入程序、命令、数据等,是人对计算机进行控制的重要工具。

计算机标准键盘的键数,少的有 83 个,多的可达 105 个,视使用要求而异。键盘可划分成三个区。中央为打字键盘区,包括字母、数字、符号和一些特殊功能键,如换档键、回车键等。键的排列和标准打字机相同,已有 ISO 2530 和国家标准 GB 2787 规定。这种排列不一定合理,但已习惯使用,约定俗成。右侧为数字键区,用于数字输入和进行四则运算,也可作屏幕编辑、移动光标使用。第三个区设置在键盘的左边或上方,是 10~12 个特殊功能键,其具体功能由使用者定义。

图 1 是个人计算机增强型键盘的键盘排列图。键盘的每个键有一个键开关,键开关有机械触点式、电容式、薄膜式等多种。其作用是检测出使用者的按键动作,把机械的位移转换成电信号,输入到计算机中去。

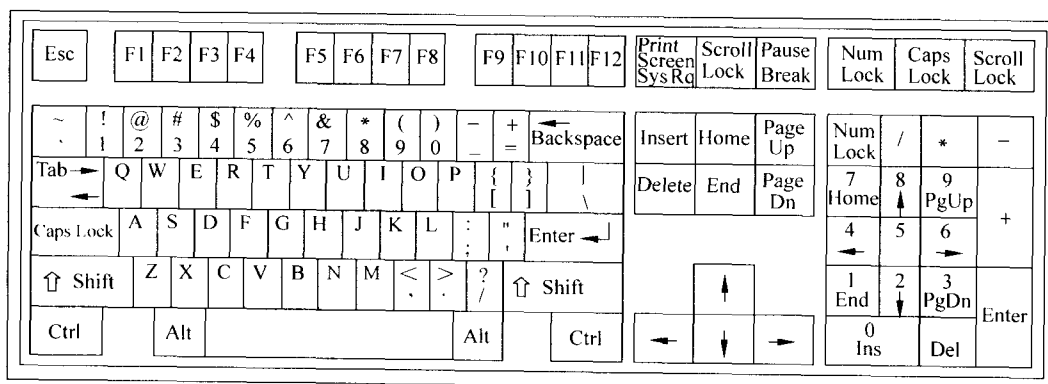


图 1 键盘排列图

键开关电路依一定的顺序排成一个 $m \times n$ 的矩阵,即每一个代表某一字符的键开关在 $m \times n$ 矩阵上有一个固定的位置。例如 IBM-PC 机的键盘,字符 a 的键位是 $m = 14, n = 2$ 。

键盘内部有键盘控制器,它由单片机和一些芯片组成。单片机进行扫描,判断使用时按键的位置,保存键的位置代码,并输入到计算机中去。

由于标准键盘功能强,也已用作汉字输入,不同的是各个键所代表的不是英文字母而是汉字的拼音字母或字形。它由各种汉字输入方法自行定义,键盘把键的位置代码送入计算机,经过软件的处理,转换成汉字的机内码,再进行其他操作处理。

台式计算机和大型计算机的终端设备所用的键盘是一种独立的部件,通过电缆和主机连接。

由键盘输入的内容通常由显示器屏幕即时显示出来,因此使用非常方便。键盘价廉、轻便。但在掌上型计算机中因键过小,影响输入速度。近年来,鼠标器、控制球等输入设备的使用不断普及,在某些方面取代了键盘,但键盘仍是人机交互式输入设备中的一个重要设备。
(林兼)

jiaocha bianyi chengxu

交叉编译程序 (cross compiler) 本身在机器 A 上运行但生成另一台机器 B 上的目标代码的编译程序。这两台机器中前一台 A 称为交叉编译程序的宿主机,后一台 B 称为交叉编译程序的目标机。

假定语言 L 是一种适合写编译程序的语言,且

在机器 A 上有 L 的一个交叉编译程序,它生成机器 B 上的目标代码,该交叉编译程序的 T 图如图 1 所示,或写为 $L_A B$ (参见自编译程序),可以利用它来实现机器 B 上的任一语言,如 FORTRAN 的编译程序。首先用 L 号机器 B 上的一个 FORTRAN 的编译程序 $FORTRAN_L B$,然后用 $L_A B$ 对 $FORTRAN_L B$ 进行编译,便可得到机器 B 上的 FORTRAN 编译程序 $FORTRAN_B B$ 如图 2 所示。

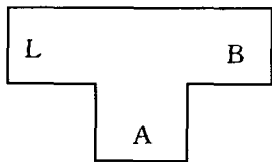


图 1 交叉编译程序的 T 图

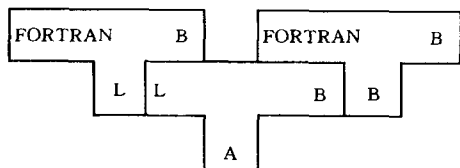


图 2 交叉编译过程一例

交叉编译技术的主要用处是可以在较大型的具备必要的软件开发工具的宿主机上为小型的或硬件配置简单的目标机开发高质量软件给予支持。交叉编译技术还可以在现有的机器上为正在制造中的新机器上的软件开发给予支持。

参考文献

Aho A V, Sethi R, Ullman J D. Compilers Principles, Techniques, and Tools. Addison-Wesley, 1986
(徐永森)

jiaohushi dianshi

交互式电视 (interactive television) 观众通过遥控器和菜单可以选择和控制节目的电视。它是一种非对称双工通信模式(参见异步传输)的新型电视业务。交互式电视分两种:节目间交互式电视和节目内交互式电视。

节目间交互式电视又称为点播电视(VOD),它分为真点播电视(TVOD)和准点播电视(NVOD)。真点播电视要求电视台(视频服务器)对每个用户的点播请求都要及时响应,允许用户对装设在信息

中心和电视台视频盘和视频带上的节目可以随意控制。准点播电视是每隔一定时间(如 10 分钟)从头播放一套节目,用户点播某个电视节目时,交换机将终端与最近将要从头开播的频道连通,用户等待时间不会超过时间间隔。

节目内交互式电视也称全交互型电视,它能够将电视台(视频服务器)对用户的请求应答即时传送给用户,传送给用户的信息内容包括视频、图像、语音、文字和数据。

交互电视可以应用在点播电视、教育、远程购物、交互式游戏、新闻点播服务等领域。交互电视的应用一般采用非对称通信模式,即下行宽带传输,上行窄带传输。应该指出的是,在这些应用中,导航和内容的无缝集成很重要。

交互电视系统核心功能应包括传输、会话、访问控制、导航与节目选择、应用启动、媒体同步链、应用控制、表现控制和用户概况等。交互电视系统主要的组成部分有视频服务器、编码器-路由器、用户请求计算机和记账计算机,以及位于用户端的机顶盒。其中,视频服务器是交互式电视系统中最关键的组成部分,它的性能直接影响服务的质量。它可以是基于个人计算机(PC)和工作站的机群结构,也可以采用大规模并行计算机结构。

作为交互电视系统的中央控制和服务部分,视频服务器负责解决大容量视频存储、节目检索和服务、高吞吐量视频传输等需求,它具有以下功能:

- (1) 请求处理 接收用户的访问请求;
- (2) 许可控制 检查用户的权限,考虑新请求的加入是否影响已有服务的性能;
- (3) 节目检索 从服务器的存储系统中检索节目的存放位置;
- (4) 可靠的流传输 向用户提供一个实时的数据流;
- (5) 支持录像机功能 如暂停、启动、快进、快退等功能。

机顶盒(STB)是一种与电视机接口并提供附加服务的设备。交互式电视的机顶盒应具有以下功能:①提供与网络的接口;②音频与视频的解码;③用户界面和图形控制;④外围设备控制;⑤安全与权限管理。
(钟玉琢)

jiaohushi yuyan

交互式语言 (interactive language) 支持使用者与计算机系统通过交互应答完成计算任务的语

言。又称会话语言。

交互式语言程序执行过程中可以请求用户输入数据、指出下一步所要采取的动作。用户也可以中止程序的执行,待完成其他工作后再从程序中断点接着往下执行。交互式语言程序执行停止时,只要系统仍处在该语言的执行环境中,则运行程序所设置的状态(变量及其所赋的值等)保持不变。

交互式语言程序一般通过解释方式执行。用户及程序人员可以在同一环境下输入、调试、运行程序,这使程序开发比较容易,程序设计环境对用户也比较友善,但也因此影响程序执行效率。

BASIC 语言是最常见的交互语言,bBASE,Fox-Base 等数据库语言实际上也是交互式语言。

参考文献

邓良松等. BASIC 语言. 北京: 中国铁道出版社, 1993 (徐宝文)

jiaohuanji

交换机 (switch) 由输入输出接口以及具有交换分组或信元等数据单元能力的转发逻辑组成的网络设备。转发逻辑描述了用于交换技术的规则,实现数据单元的转发。输入输出端口是物理的和逻辑的接口,用来连接到需要交换数据单元的通信网络。

交换机的操作和网桥的操作相同,转发逻辑对网上的设备是透明的,交换机的逻辑模块如图 1 所示。

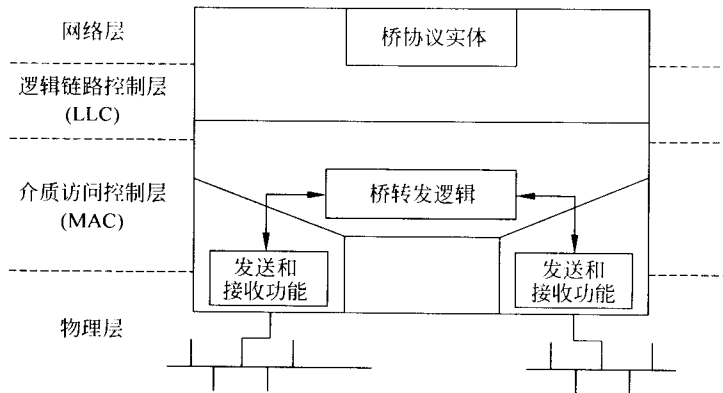


图 1 交换机的逻辑模块

图 1 包括了 OSI 模型的下 3 层,但基本的操作是在第 2 层,即介质访问控制 (MAC) 层分组的交换。第 2 层交换机的转发逻辑如图 2 所示。图中包括了转发逻辑的 3 个关键组成: 滤波与转发查找逻辑、学习逻辑和源地址表 (SAT)。

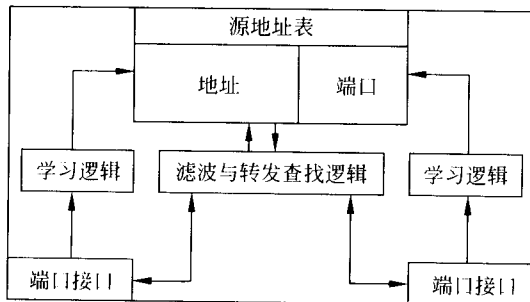


图 2 交换机的转发逻辑

交换机的转发逻辑有两个十分重要的特点: 透明操作和即插即用操作。转发逻辑的透明操作是基于这样一个原则,转发第 2 层分组时,不论是传递或被淹没,都不需要修改 MAC 层的目的站,因而也就对端系统没有任务配置的要求。交换机的第 2 个特点是即插即用配置。因为转换逻辑的学习逻辑自动提供源地址表,不需要对网上的交换机作任何配置。事实上,交换机在网上存在愈久,则学习逻辑收集的信息愈多,它的智能也愈强。

交换机的第 2 个重要组成是 I/O 端口。因为第 2 层网络有很多类型,因此需要很多不同的物理接口用于交换机,通常有两个接口规则交换机必须支持: 访问端口和网络上连端口。作为网络上连端口,它必须具有足够的频宽以转发来自多个交换端口的通信量。访问端口必须用正确的技术将交换连到端系统或共享的段。网络上连端口和访问端口的

概念完全是逻辑的。对端口的定义需要一个已知的端口号以便正确地使用交换机。

有两种交换方式用于交换机,一种是存储转发方式,另一种是穿通方式。传统的交换机采用转发逻辑,即存储转发方式。采用存储转发方式时,交换机在转发分组至目的段以前,接收并缓存整个分组。这样可使交换机在传递分组至目的段以前识别并丢掉有差错的分组。这个方法还可互连不同技术的局域网(LAN),因为当缓存整个分组后,还可对不同MAC的LAN进行转换。

穿通方式是另一种交换机转发逻辑。其目的是减少由于存储转发逻辑引入的延迟。穿通方式不需缓存整个分组,而只需缓存用以决定目的地址的帧头,这样在任何时候只有很小一部分分组被缓存在交换机。通常穿通方式交换机用于解决小的工作组网络,没有上连的需求。这种交换机很便宜,因而不需大容量的缓冲存储器。大规模的交换机LAN,主要的交换机仍采用存储转发交换机,因为它可提供必要的差错隔离和上连能力,从而可构造一个可靠的、可扩展的交换机LAN。

第3层交换机通常定义为具有转发带有第3层及3层以上协议知识的数据分组能力的设备。第3层交换机有3种类型:路由器、嵌入路由器和混合型第3层交换机。

第1类第3层交换机是传统的路由器。通常,路由器比交换机性能要差得多,这种作为第3层交换机的路由器要比传统的路由器更快,但基本操作原理是相同的。

第2类第3层交换机是具有嵌入路由器功能的LAN交换机。事实上这种类型的交换机只是将LAN交换机和路由器放在一个机箱内,这个设备既是LAN交换机,又是路由器。

第3类第3层交换机是在第3层交换机基础上使用某些第3层知识作转发决定的交换机。这类交换机的关键组成是所有通信在基于MAC层地址的第2层进行。它的第3层服务是基于了解某些MAC广播的能力作为它的第3层功能。这样它能利用包含在分组第3层的部分信息,以决定分组发送至何处。作为这类交换机的一个例子是能够识别IP协议,以及能知道端系统和IP网络的第3层地址的交换机。

参考文献

1. 胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002

2. Roesse J. Switched LANs. McGraw-Hill Companies Inc., 1999 (胡道元)

jiaohuan jishu

交换技术 (switching technologies) 采用交换机或接点机设备和有限线对,通过路由选择,在要求通信的双方之间建立物理的或逻辑的连接,形成通信电路,以实现通信双方语音或数据传输的技术。交换技术起源于电话交换系统,一直到20世纪70年代,当人们说到交换,指的仍然是电话交换。计算机网络的兴起和发展,就有了数字交换的概念和一系列飞速发展的交换技术。现在通常采用的交换技术有:电路交换、存储转发交换、分组交换、帧中继交换和异步传送模式等。

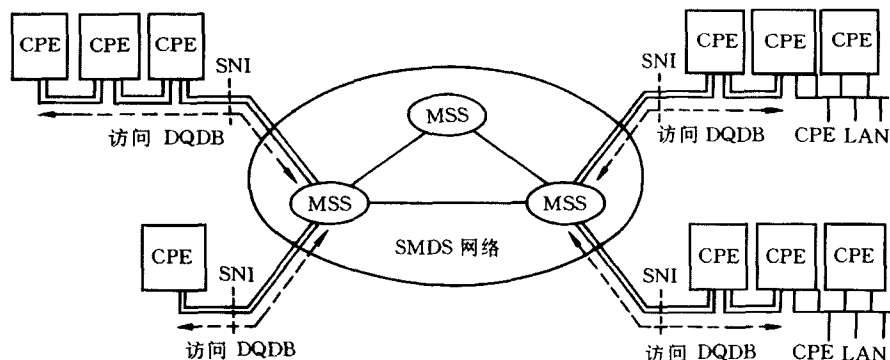
(史美林)

jiaohuanshi duozhaowei shuju yewu

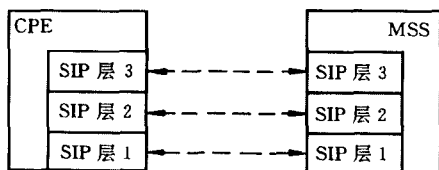
交换式多兆位数据业务 (switched multi-megabit data service, SMDS) 为满足用户的宽带广域连接的要求而提供的一种高速、公用的分组交换数据服务。它可以在城市区域范围内提供局域网那样的高速连网性能,传输速率从T1(1.544 Mb/s)到T3(45 Mb/s)。SMDS是基于贝尔通信研究所提出的标准,结合城域网MAN的标准(IEEE 802.6)发展起来的,用于网络互连,提供无连接数据报服务,能高容量、低延迟、可靠地传输数据报。在用户-网络接口的用户侧使用分布式队列双总线(参见城域网)接入。以后,SMDS可以提供传输速率为155 Mb/s的与同步光纤网相连接的服务。

SMDS基本结构如图1所示。

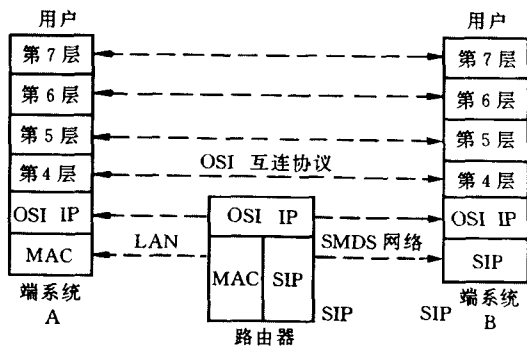
图中,SMDS传输交换网络由城域网交换系统(MSS)构成。当宽带综合业务数字网技术成熟后,可用它完成SMDS的功能。用户办公室设备(CPE)通过连接在用户网络接口(SNI)上的DQDB访问SMDS,所采用的SMDS接口协议(SIP)如图2所示。SIP第三层执行管理功能,它把SMDS服务数据单元(SDU)添加上首部 and 尾部、源和目的SNI地址、检错纠错码等信息,形成第三层的协议数据单元(PDU)。SIP第二层包括按位差错检测,它将第三层的PDU进行分段、合段和成帧等。SIP第一层提供物理层的功能,如跨物理设施上的数据位串的传输,传输速率为T1和T3。



SNI——用户网络接口； MSS——城域网交换系统；
CPE——用户办公室设备； CPE LAN——用户局域网



(a) SIP 协议栈



(b) SIP 协议栈与 OSI 参考模型比较

SIP 的这三层与宽带综合业务数字网的下三层有一定的对应关系。发展基于宽带综合业务数字网的 SMDS 受到重视。

参考文献

1. 张元等编译. 从局域网到广域网——九十年代的网络管理技术. 北京: 北京市新闻出版局, 北京希望电脑公司, 1991
2. Stalling W. Local and Metropolitan Area Net-

work. Fourth Edition. New York: MacMillan Publishing Company, 1993 (史美林)

(史美林)

jiēkǒu dīngyì yuán

接口定义语言 (interface definition language) 描述对象与其外部世界接口的语言。接口是服务的提供方和使用方之间互相合作的约定, 它仅描述如何使用的信息, 不包括具体实现信息, 接口体现了“信息隐蔽”的原则。

接口定义语言与编程人员通常使用的**程序设计语言**都是计算机语言。接口定义语言独立于具体的程序设计语言,但可以向它自动映射,例如:生成相应的类型说明,生成相应的代理机制等。这对于快速开发客户程序十分有利。

接口定义的核心是对操作的描述。通常将对操作的一个形式化描述称为操作的标记。随着网络软件的发展,对操作描述的内容也在逐步完善。最简单的操作仅包括功能性信息,复杂的操作还包括约束性信息。

(1) 功能性信息 接口的功能性信息是对接口中各个操作的功能描述。操作是由操作符标识的实体,它指明了一个不可再分的服务原语。与数学上一个函数的映射过程类似,对一个操作功能的描述主要由输入、输出两部分组成,分别用于描述操作的输入、输出参数名称及类型。例如:操作标记 `int add(in int a 1, in int a 2)` 定义了一个加法操作,该操作的输入是两个整数,输出是一个整数。

(2) 约束性信息 接口的约束性信息是指对功

能约束的描述。网络环境下不同软件模块之间的合作需要考虑的因素不仅包含功能方面,还涉及分布性、可靠性、安全性等方面的因素。只有了解了这些因素,不同的软件才能真正合作起来。因此,网络环境下的软件接口除了需要定义模块的功能性内容外,还需要定义模块的约束性内容。简单的约束包括:异常处理、时间限制、客户身份、可靠性要求等。复杂的约束包括:①行为特征,用于描述操作的外部特征,这一般是通过的操作增加前置与后置条件而实现的;②同步特征,用于描述操作的并发性等特征。

OMG 组织提出的接口定义语言 (OMG-IDL) 是较早出现的接口定义语言。OMG-IDL 于 1991 年提出,并且已经被 ISO 组织采纳 (ISO DIS 14750)。后来陆续发展的接口定义语言还有:微软的 MIDL, Web 服务的 WSDL 等。

由于接口较好地分离了相互协作的软件,使它们可以分别演化,比较好适应变化,因此受到了软件研究人员的广泛关注。

参考文献

OMG. The Common Object Request Broker: Architecture and Specification, 2.3.1. 1999 (王千祥)

jiegouhua buxian xitong

结构化布线系统 (structured cabling system, SCS) 一种模块化、灵活性极高的建筑物和建筑

群内的信息传输系统。结构化布线系统 (SCS) 是一种集成化的通用传输系统,它利用双绞线电缆或光缆来传输建筑物内的多种信息。

在现代化的大型建筑中,除计算机网络布线系统以外,通常还会有电话系统、楼宇控制系统等各专业布线系统。传统的做法是,为不同的专业系统配置不同的线缆、插座及接头等来构成各自网络。由于连接这些不同网络的插头、插座及配线架互不兼容,因此只要变动终端机的位置,就必须重新布放新的线缆、安装新的插座。在这种传统的布线方式下,因办公室的重新规划及办公设备的变更而导致布线系统的变更要耗费大量的金钱及时间。同时,传统的布线方式对于布线系统的日常维护和管理、故障的检查和排除都不是很方便。

为解决传统布线方式中的种种弊端,工业界推出了结构化布线系统。SCS 将所有的话音、数据、图像及监控设备的布线组合在一套标准的布线系统上,采用统一的线缆、插头、插座及配线架。当终端机的位置需要变动时,只需将其插入新地点的插座上,然后作一些简单的跳线就行了,不需要再布放新的线缆,也不需要安装新的插孔。另一方面,SCS 采用星型结构,系统的管理维护及故障的检查和排除也非常方便。SCS 以其高度的灵活性及多元化服务而越来越得到人们的重视。

结构化布线系统 SCS 可以划分为 6 个子系统,如图 1 所示。

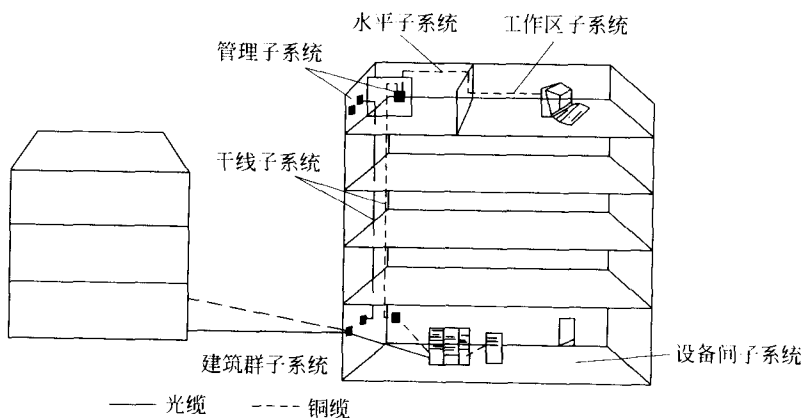


图 1 结构化布线系统的组成

组成结构化布线系统的 6 个子系统为: ①工作区子系统 (用户端子); ②水平子系统; ③干线子系统

系统,也叫垂直子系统; ④设备间子系统; ⑤管理子系统; ⑥建筑群子系统。

参考文献

胡道元主编. 网络设计师教程. 北京: 清华大学出版社, 2001 (王晓东)

jiegouhua chengxu sheji

结构化程序设计 (structured programming) 具有结构性的编程方法。结构性主要反映如下: 第一, 编程工作为一演化过程, 即按抽象级别依次降低、逐步精化, 最终得出所需程序的方法编程(自顶向下逐步精化); 第二, 按模块组装的方法编程, 亦即, 将所需程序编制成由若干模块(或构件)组成; 第三, 将所需程序编制成只含顺序构造、判定构造以及重复构造, 其中每种构造只允许单入口和单出口。

采用结构化程序设计方法编程, 旨在提高编程质量与所编程序的质量。自顶向下、逐步精化方法有利于在每一抽象级别上尽可能保证编程工作与所编程序的正确性; 按模块组装方法编程以及所编程序只含顺序、判定、重复三种构造则可使程序结构良好、易读、易理解、易维护, 并易于保证及验证程序的正确性。

1964 年 C. Bohm 与 G. Jacopini 曾证明, 任一流程图均可利用重复与嵌套等价地改写成只含可执行语句列、判定子句和迭代构造, 并且每种构造只有一个入口和一个出口。1968 年 E. W. Dijkstra 在给 ACM 通讯的一封信(该信由 ACM 编辑添上如下标题: 转向语句视为有害)中指出, 据彼观察, 程序片的易读性与易理解性与其中所含之无条件转移控制(转向)个数成反比。该信发表后, 引起普遍重视, 结构化程序设计方法逐渐形成, 并成为程序设计领域、计算机软件领域的重要方法, 对计算机软件之发展意义重大, 作用明显, 相应出现了诸如 Modula-2, C, Ada 等所谓结构化程序设计语言。

结构化程序设计方法之主要贡献有二: 一是将程序设计方法由技艺向科学迈进一步; 二是据此方法所编出之结构化程序不只是供计算机阅读而且也是供人阅读的创造性工作。

参考文献

1. Bohm C and Jacopini G. Flow Diagrams, Turing Machines, and Languages with Only Two Formation Rules. Comm. ACM. , 1964, 9(5)
2. Dijkstra E W. Go to Statement Considered Harmful. Comm. ACM. , 1968, 11(3)
3. Dahl O J, Dijkstra E W and Hoare C A R.

Structured Programming. New York: Academic Press, 1972 (徐家福 仲萃豪)

jiegouhua fangfa

结构化方法 (structured method) 强调开发方法的结构合理性以及所开发软件的结构合理性的软件开发方法。结构是指系统内各组成要素之间的相互联系, 相互作用的框架。结构化开发方法提出了一组提高软件结构合理性的准则, 如分解与抽象、模块独立性、信息隐蔽等。针对软件生存周期各个不同的阶段, 它有结构化分析(SA)、结构化设计(SD)和结构化程序设计(SP)等方法。

结构化分析方法给出一组帮助系统分析人员产生功能规约的原理与技术。它一般利用图形表达用户需求, 使用的手段主要有数据流图(DFD)、数据词典(DD)、结构化语言、判定表及判定树等。数据流图以图形的方式表达目标系统中信息的变换和传递过程, 有以下 4 个基本要素: 数据流、加工、文件、数据源和数据宿。数据流由一组固定成分的数据组成, 它具有名字和流向; 加工是对数据流的变换; 文件是可访问的存储信息; 数据源和数据宿是存在于计算机系统之外的实体, 分别表明数据处理过程的数据来源及数据去向。SA 方法采用分层的数据流图来表达一个目标数据处理系统, 在保证一致性及完备性的情况下, 采用数据流图分解及抽象的手段, 来控制需求分析工作的复杂性。

与数据流图配合使用的是数据词典, 它对数据流图中出现的所有数据元素给出逻辑定义, 用以对数据流图中的各要素得到确切的解释。

在分层数据流图中, 最底层的数据加工(称为基元)不需再用下层 DFD 进一步描述, 而可采用结构化语言、判定表或判定树等描述其功能。结构化语言有基本的过程控制结构, 如顺序、选择和重复, 可对 DFD 中的功能进行过程化描述。

结构化分析的步骤如下: ①分析当前情况, 作出反映当前物理模型的 DFD; ②推导出等价的逻辑模型的 DFD; ③设计新的逻辑系统, 生成数据词典和基元描述; ④建立人机接口, 提出可供选择的的目标系统物理模型的 DFD; ⑤确定各种方案的成本和风险等级, 据此对各种方案进行分析; ⑥选择一种方案; ⑦建立完整的需求规约。

结构化设计方法给出一组帮助设计人员在模块层次上区分设计质量的原理与技术。它通常与结构化分析方法衔接起来使用, 以数据流图为基础得到

软件的模块结构。SD 方法尤其适用于变换型结构和事务型结构的目标系统。在设计过程中,它从整个程序的结构出发,利用模块结构图表达程序模块之间的关系。

模块结构图是采用结构化设计方法进行软件概要设计的重要描述手段。它以图形的形式描述软件系统的模块组成及模块之间的调用关系。

构成模块结构图的主要成分有模块、调用和数据。在结构图中的模块以矩形表示,在矩形框内可以标以模块的名字。模块间如有箭头或直线相连,表明它们之间有调用关系。对于两个处在不同位置上的模块,通常把上面的称为调用模块,下面的称为被调用模块。调用箭头上的小箭头表示模块调用时模块间的数据传送,小箭头的方向指明了传送的方向。数据也应用适当的名字来标识。

结构化设计的步骤如下:①评审和细化数据流图;②确定数据流图的类型;③把数据流图映射到软件模块结构,设计出模块结构的上层;④基于数据流图逐步分解高层模块,设计中下层模块;⑤对软件模块结构进行优化,得到更为合理的软件结构;⑥描述模块接口。

参考文献

1. DeMarco T. Structured Analysis and System Specification. New York: Yourdon Press, 1979
2. Yourdon E. Managing the System Life Cycle. New York: Yourdon Press, 1982 (吕建国 钱乐秋)

jiiegouhua fenxi yu sheji jishu

结构化分析与设计技术 (structured analysis and design technique, SADT) 由一组称为结构分析图的图形语言工具与使用这组工具的方法、管理技术所构成的系统分析与设计。SADT 的结构分析图用于构成 SADT 模型来表达需求分析的结果。SADT 方法自顶向下地建立 SADT 模型。

SADT 模型由一组有序的结构分析图构成,每张结构分析图是由若干个结点以及连接这些结点的弧组成的工程图。SADT 模型是一种分层的系统模型,它自顶向下地把高层的结点分解成为若干个下层图。

SADT 模型有两种基本类型:以描述系统中的活动为主的活动模型和以描述系统中的实体为主的“数据模型”,分别用**活动图**和**数据图**表示。活动图的结点表示活动,弧表示活动之间的数据流;数据图

表示结点内数据对象及其弧上的活动。因而,数据图和活动图是对偶的,活动图和数据图统称为结构分析图。一个完整的 SADT 模型是由多个视角的活动模型和数据模型构成。在实践中,活动图应用更广泛。

图 1 给出了活动图和数据图中结点的一般形式,每个结点有四种不同类型的弧,结点左边进来的弧是输入,结点右边引出的弧是输出,顶端进入的弧称为控制,而底部进入的弧称为机构。在图中,每个结点用四种类型的弧与其他结点发生联系。一个结点的输出必定是另一个结点的输入或控制,或者是整个系统的输出。每个结点的输入或控制也必定是另一个结点的输出或外部环境的输入。

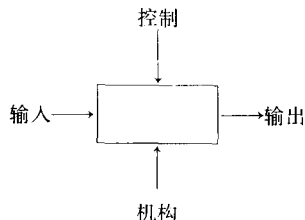


图 1 基本结点图

在活动图中,输入和控制表示完成此结点的活动所需要的数据,输出表示执行该活动所产生的结果数据,机构表示完成该活动的机构(人或设备)。在数据图中,输入是生成数据对象的活动,输出是使用数据对象的活动,控制是对结点活动条件的一种约束,机构是用来储存数据对象的设备。在活动图中的活动和数据图中的数据,均可进一步细化,形成一个层次的活动图或数据图,从而形成一个由多层次的结构分析图所构成的系统模型。

SADT 首先是由 T. D. Ross 在 Softech 公司从事大型软件开发的实践中,总结出来的一种通用的处理复杂问题的方法。1977 年由 Ross 等人发表的著名论文“需求定义的结构化分析”奠定了这一方法的基础。70 年代中后期, SADT 成为最有代表性的结构化软件开发方法之一。该方法被广泛地用于系统的定义、软件需求分析、系统设计以及软件设计,特别适于规模较大和复杂的系统的分析与设计,曾被 ITT 作为软件规约与设计标准。这一方法和技术中的活动图模型被美国空军公布的 ICAM 工程所采纳,并作为其 IDEF (ICAM definition method) 软件开发方法的 IDEFO 模型,得到了广泛的应用和推广。

(郝克刚 王斌君)

jieshi chengxu

解释程序 (interpreter) 按照源程序中指令或语句的动态执行顺序,逐条翻译,并立即解释执行相应功能的处理系统。

解释程序的突出特点是不把源程序翻译成机器语言形式的目标程序,而是直接将源程序中的指令或语句转换成加工数据的动作或完成所需功能的动作。

解释程序是由总控程序和一组相应各种类型指令或语句的执行子程序组成(图1)。其工作过程如下:首先,由总控程序完成初始化工作,将系统置成初态;然后,依次从源程序中取出一条指令或语句,并进行语法检查,如果语法有错,则输出错误信息;如果语法正确,则根据指令或语句的类型,转去执行相应类型的执行子程序;自执行子程序返回后,检查源程序是否解释完毕,如果未完成,则继续解释执行下一条指令或语句;如果完成,则由总控程序完成必要的善后处理工作。

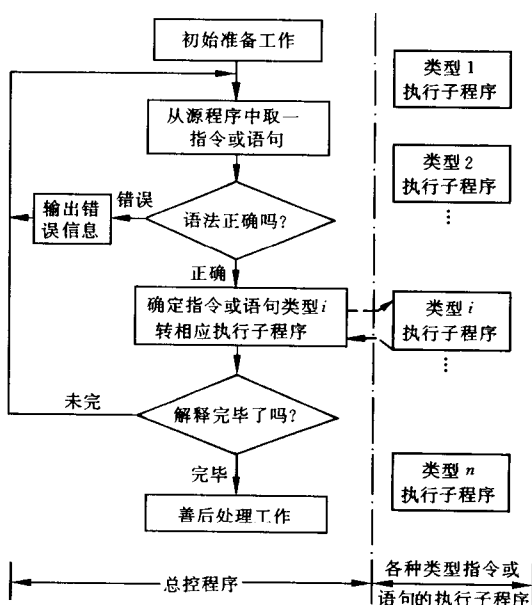


图1 解释程序的结构和工作流程示意图

解释程序的优点是实现算法简单,且易于在解释过程中灵活、方便地插入所需的修改和调试措施;其缺点是运行效率低。例如,对于源程序中需要多次重复执行的指令或语句,解释程序将要反复地取出、翻译和解释执行它们。根据这些特点,解释程序

通常适用于交互方式工作的,或调试状态下运行的,或运行时间与解释时间相差不大等类型的语言。随着超大规模集成电路的迅速发展,人们提出了诸如FORTRAN机、COBOL机之类的高级语言机器的概念。其基本思想就是采用软件固化的方法,通过微程序设计语言实现高级语言的解释程序。伴随计算机硬件和软件技术的不断发展,解释程序将有广阔的应用前景和发展前途。(曹东启)

jieshi jizhi

解释机制 (explanation mechanism) 当用户对专家系统的推理提出疑问时,由该专家系统给出的回答机制。

在实用的问题中,许多领域专家如医生、企业决策者、军事战略家等在使用专家系统作出结论或决策时,除非系统能对所产生结论的推理过程作出详细说明,以说服用户相信它所作的结论是合理的,否则它很难使用户信服和承认;或者由于用户不敢承担由采用专家系统的结论所带来的风险,而很难接受专家系统的推论。因此,专家系统的解释机制能增强用户对它的信任程度。

对一个解释程序有下述几点基本要求:①能解释有关系统的知识和行为方面的问题;②对用户提出的问题能给出一个正确的和易于理解的回答;③使用户易于使用。

解释程序一般能回答的问题有:

(1) 推理状态检查 回答系统运行过程中有关推理的问题,常见的这类问题有三种:回顾性推理问题解释系统是如何达到这个特定状态的,例如显示出得出结论的推理链;假设性推理问题解释如果某些特殊的事实或者规则改变的话,将会发生什么变化;矛盾性推理问题解释为什么一个所期望的结论没有达到。

(2) 一般性问题回答 回答咨询中有关系统知识库的当前状态的问题。

解释机制除了解释推理的路径和理解知识库的内容这一个主要功能之外,还带来一些辅助的功能。诸如在系统维护时,可帮助知识工程师发现和更正知识库中的知识错误;在追踪推理路径时,发现推理的不一致性;对初级用户起到一定的辅助教学的作用。这是因为专家系统中知识能有效地支持高质量的专家决策,通过解释可以使用户了解专家决策的过程,从而获得这些知识,形成一个进行教学的过程。

参考文献

Rich E A 著. 人工智能引论. 李卫华, 汤怡群, 文中坚译. 广州: 广东科学技术出版社, 1986

(施鸿宝)

jiemian gongju

界面工具 (interface tool) 用来创建和管理软件用户界面的软件。又称用户界面管理工具 (UIMT) 或用户界面开发工具 (UIDT)。

交互式软件需要与人进行大量通信, 用户界面就是软件与用户之间的联系纽带。采用用户界面工具, 不仅可以加速软件用户界面的开发速度, 而且能把软件开发工作中的功能部分开发工作和用户界面开发工作分开。

一个交互式软件通常由三部分组成, 即: 功能部分、用户界面部分和对话接口。功能部分专门用来实现软件的基本功能, 用户界面部分接受用户输入并显示处理结果, 对话接口用于在前两者之间进行信息交换。这样, 功能部分和用户界面软件相互分离、各自独立。

用户界面软件由表示部分、布局部分和对话部分组成。

表示部分管理各种输入装置 (如键盘、鼠标、触摸屏、记录笔等) 的驱动程序、显示屏幕上的界面对象或进行其他类型的输出 (如语音合成等)。

布局部分确定可用的一组界面对象中哪些已被选择以及它们应该如何排列组合。它还规定了每个对象的属性值, 例如对象的颜色、字体、大小和位置等。布局软件能定义构成界面的对象层次, 高层对象可包含低层对象, 低层对象能继承高层对象的特性, 这使得定义用户界面更加方便并能复用。

对话部分描述了功能部分对用户输入的响应, 指明什么时候该显示从功能部分送来的数据, 并提示用户输入信息, 即在用户界面软件和功能部分之间传送控制信息。

用户界面工具主要有窗口库函数和工具箱、布局语言、交互设计工具 (IDT) 等。

窗口库函数和工具箱 窗口系统, 如 X Window System (Scheifer, Gettys, 1986), Microsoft Windows (Petzold, 1990) 等, 提供了具有基本窗口管理功能 (如移动窗口、改变窗口大小、在窗口中输入输出数据等) 的库函数和一个工具箱, 用来开发用户界面软件的表示部分。**工具箱** 包含了各种公用的用户界面对象, 如按钮、检测框、标题域、滚动条等以及管理这些对象的软件。X Window System 的开发者把界

面对象称为 widget。利用工具箱中的界面对象可以使开发的用户界面具有一致的风格, 而且能使用用户界面编程更加容易。工具箱的例子有 X Toolkit (McCormack, Asente, 1988) 和 Next Step (NeXT, Inc, 1989) 等。

布局语言 布局语言支持用户界面的布局, 它以正文的形式描述界面的静态表示。用布局语言描述的界面布局称为资源文件, 它要通过专门的编译程序进行编译才起作用。资源文件里包含有 widget 属性的组合方式、属性值以及其他程序员定义的信息。两个著名的布局语言是 OSF/Motif (1989) 的用户界面语言 UIL 以及 Wel (1990)。

交互设计工具 交互设计工具 (IDT) 允许设计者采用交互方式定义用户界面的布局。某些 IDT 提供了具有类似于图形编辑功能的直接操作编辑程序, 可以用鼠标来放置 widget 或改变其大小。用户界面设计者还能在 IDT 提供的表格中填入指定的属性和事件驱动的回调函数名字。

用户界面设计者使用 IDT 产生布局定义和相关函数的框架代码。框架代码以程序设计语言 (如 C 或 Ada)、资源文件或 Motif 的用户界面语言 (UIL) 等形式给出, 然后应用程序员再把功能部分函数加到框架代码中。

用户界面工具自 20 世纪 80 年代初以来一直十分活跃。早期的系统有 GWUIMS (1985), Menulay (1983) 和 FLAIR (1982)。随后出现了 Machintosh 环境上的流行工具 HyperCard (1987) 和 SuperCard (1989), NeXT 公司的 NeXT Interface Builder (1989), X Window System (1986), Sun Microsystem 公司的 OpenLook (1989), 以及成为目前工业标准的 Open Software Foundation Inc. 的 Motif (1989) 和 Microsoft 公司的 Windows (1990) 等。我国自 80 年代中期开始这方面的研究, 陆续推出了一些实用系统。

参考文献

1. Hix D. Generations of User-Interface Management Systems. IEEE Software, Sept., 1990
2. Lee E. User-Interface Development Tools. IEEE Software, May., 1990: 31 ~ 36
3. Myers B A. User-Interface Tools: Introduction and Survey. IEEE Software, Jan., 1989: 15 ~ 23

(金茂忠)

jincheng

进程 (process) 程序的一次执行过程。反过来看, 程序则是进程的一种静态描述。进程是在操作

系统的管理下被启动,进入运行状态,并在一定条件下中止或结束的。进程的运行需要使用一定的计算机资源(处理器、内存、各种外部设备等)。因此,进程又是操作系统调度的基本单位。

一个普通(非并发)的程序被执行时,相当于产生了一个新的进程;执行完毕后,这个进程就消亡了。为了提高程序的执行速度,可以把一个程序设计成多个可以“同时”执行的部分,这就是所谓的**并发程序**。执行并发程序时,可以有多个对应于此程序的进程同时处于运行状态。其中不同的进程如果在某一瞬间使用的是不同种类的资源,或者虽然是同一类,但是系统拥有多台这类设备(例如,多处理器系统中的多台处理器),那么这时这些进程是真正在物理上“同时”运行的。反之,它们就必须以某种次序轮番运行。但我们认为它们在逻辑上仍然是可以“同时运行”的。

然而,除了资源使用上的冲突之外,并发程序中的多个进程之间并非完全独立无关,它们之间往往需要相互合作,才能协同完成任务。例如进程 A 在运行到某一阶段后,必须等待进程 B 向它提交某种数据,然后才能继续运行,这就是一种通信关系。还有一种是“互斥关系”,例如,有一个飞机订票系统,为每一订票终端都安排了一个对应的进程。每当有一个旅客在某一个窗口订票时,相应的进程便被启动,去查看是否有空位。如果有,就出售与该座位对应的票,并修改数据,以免该票被重复出售。但是,由于不同售票窗口对应的进程是可以“同时运行”的,因此有可能出现两个进程同时查看同一座位的情况:两个进程同时发现同一座位有空。于是仍然会出现一张票卖给两位旅客的错误。为了避免这一点,只需规定“查看是否有空位”的这一段程序在同一时刻最多只能让一个进程进入。

并行程序中的多个进程不一定是在一开始就同时出现并一直维持到整个程序结束,而是可以根据需要动态地产生和消亡。换言之,一个进程可以在运行过程中派生出新的进程来。新派生出来的进程又可以继续派生下一代的进程,于是形成了进程之间的“父子关系”。当一个新的子进程诞生时,它所需要的资源通常是由其父进程的资源中划分而来。不同的进程在互相独立的内存空间中运行,这一点是和“线程”不同的。同一进程内又可以分解为若干线程,这些线程在逻辑上也是可以同时运行的,但是它们共享同一内存空间,因此线程之间的调度无需操作系统的介入,从而减小了系统开销、提高了

效率。

(周锡令)

jincheng daishu

进程代数 (process algebra) 关于通信并发系统的代数理论的统称。

20 世纪 70 年代后期,英国学者 R. Milner 和 C. A. R. Hoare 分别提出了**通信系统演算**和**通信顺序进程**,开创了用代数方法研究通信并发系统的先河。此后这一研究方向兴盛不衰,出现了众多类似而又相互区别的演算系统,如 ACP(提出者 J. A. Bergstra 和 J. W. Klop),ATP(提出者 M. Hennessy),Meije(提出者 G. Boudol, R. de Simone),LOTOS 等,统称为进程代数。这些代数理论都使用通信,而不是共享存储,作为进程之间相互作用的基本手段,表现出面向分布式系统的特征。

在语法上,进程代数用一组算子作为进程的构件。算子的语义通常用结构化操作语义方法定义,这样进程就可看成是带标号的变迁系统。进程代数的一个显著特征是把并发性归结为非确定性,将并发执行的进程的行为看成是各单个进程的行为的所有可能的交错合成,即所谓交错语义。

进程代数研究的核心问题是进程的等价性,即在什么意义下两个进程的行为相同?在进程代数领域使用的最为广泛的等价关系有互模拟、测试等价、失败等价(参见**通信顺序进程**)等。对这些语义等价关系均建立了相应的公理系统。关于公理系统的研究不仅加深了对语义理论的理解,而且使得有可能对语义等价关系进行形式推理。

为了将进程代数的理论成果应用于解决实际问题,20 世纪 80 年代后期出现了许多计算机支持工具。用这些工具可对进程的行为进行推理或模拟。

参考文献

Milner R. Operational and Algebraic Semantics of Concurrent Processes. Chapter 19 of Handbook of Theoretical Computer Science. van Leeuwen ed. Elsevier Science Publisher B. V., 1990

(林惠民)

jingjian zhilingji jisuanji

精简指令集计算机 (reduced instruction set computer, RISC) 采用简化了的指令系统和硬连线控制器的计算机。RISC 是在高效的流水线技术(参见**计算机流水线**)的基础上充分利用指令并行执行和编译优化技术的计算机。

20 世纪 80 年代初, RISC 这一词刚刚问世时, 它的含义是简化指令系统的计算机, 它舍弃不常用的复杂指令, 并充分改进频繁使用的基本指令的实际执行效率, 把微程序控制器改为硬连线控制器, 加强寄存器-寄存器操作指令, 从而简化了计算机结构, 提高了性能。后来, RISC 技术强调优化流水线设计, 在这个基础上使 1 条基本指令的执行尽可能地在一个机器周期内完成。但商品化的 RISC 产品并不太追求指令系统和硬件结构的简化, 而十分注重编译的优化, 使硬件与软件设计密切结合, 共同承担计算机性能提高的任务。

RISC 的发展历程

(1) RISC 的问世 在 20 世纪 70 年代中后期, 不少学者研究计算机指令系统的实际执行效率, 他们通过统计和分析, 研究指令系统中究竟哪些指令的执行效率是高的? 各种指令的使用频度如何? 哪些指令是不常用的或执行效率是很低的? 通过这些研究, 得出著名的“20% / 80%”定律。该定律表明, 在当时大部分计算机的指令系统中, 只有约 20% 的指令是经常使用的, 它们约占程序执行总指令数的 80%。而指令系统中的约 80% 的指令在实际上是很少使用的, 它们一般只占程序执行总指令数的 20%。70 年代中后期, IBM 公司采用简化指令系统思想设计了 IBM 801, 即对于常用的基本指令做优化设计, 注重提高指令执行效率, 而对于不常用的复杂指令, 则尽量少用或不用, 或靠例行子程序来完成。应该说, IBM 801 是 RISC 思想的最早实践。

超大规模集成电路 (VLSI) 工艺在 80 年代初取得了很大的发展, 在处理器芯片上设置含有较多寄存器的通用寄存器堆已经很容易实现。这有利于使用寄存器-寄存器操作来实现大多数基本指令, 如加法、减法和逻辑操作。操作数都取自芯片上的寄存器堆, 以加速基本指令操作的执行, 而且操作数地址字段只需指明寄存器在堆中的编号, 所以地址字段也较短, 有利于在一条 32 位字长的指令中包含 2 个源操作数地址和 1 个操作结果地址。这样便于实现固定格式和固定字长的指令字设计。80 年代初, 美国加州大学伯克莱分校研制了单芯片的精简指令集计算机, 定名为 U. C. B. RISC I 和 RISC II, RISC 的名字正式问世。与此同时, 斯坦福大学也研制出 RISC 类型的 MIPS 芯片。U. C. B. RISC II 芯片使用了 3 万多个晶体管, MIPS 芯片使用了 2.5 万个晶体管, 它们的运行速度都超过了当时著名的超级小型计算机 VAX-11 / 780, 这在实践中证明了 RISC 思想

的正确性。

(2) RISC 产业的形成 20 世纪 80 年代中后期, SUN 公司和 HP 公司在使 RISC 真正成为工业产品, 并且在市场上畅销。SUN 公司在 U. C. B. RISC II 基础上设计了 RISC 产品 SPARC 微处理器, 并且在其工作站中取代了 MC 68020。HP 公司也研制了 HP-PA 的 RISC 产品系列。当时, RISC 在从实验室走向工业生产过程中, 要解决的重要问题是保持应用软件的兼容性。SUN 公司和 HP 公司都在 UNIX 或类似 UNIX 的操作系统的基础上, 在 C 语言源程序这一级上做到软件兼容。并花了很多力量发展 C 语言的编译优化技术, 使 RISC 的流水线执行效率提高, 从而进一步改善 RISC 工作站的性能。他们的 RISC 工作站的性能都大大超过了同时期的 CISC 工作站的性能。SUN 公司和 HP 公司在 RISC 技术上的成功, 使得 80 年代末期几乎所有公司的工作站都采用了 RISC 技术, 并采用 UNIX 或类似 UNIX 的操作系统。如 IBM 公司的 RS / 6000, SGI 公司的基于 MIPS R 2000, R 3000 的工作站等。RISC 结构和 UNIX 操作系统成为工作站的标准平台。

在微型计算机领域内, 在 80 年代中后期, Intel 公司的 CISC 产品 80286 和 80386 还是占绝对优势的, 主要是因为 80x86 / MS-DOS 平台上开发的应用软件已有上亿个。为了保持与这么多的应用软件兼容, Intel 公司采取了逐渐向 RISC 技术过渡的策略。它在 80486 和 Pentium 中吸取了 RISC 的思想, 即尽量使基本指令在一个机器周期内执行, 并且减少复杂指令的执行周期数, 同时仍保持 MS-DOS 与 Windows 环境下的应用软件的兼容性。1993 年底, IBM 公司、Apple 公司与 Motorola 公司联合开发了 RISC 微处理器 Power 601 以及以后的 Power 60x, 这使微型计算机走向 RISC 技术的总趋势更加明显。

总之, RISC 的结构是计算机体系结构的一次重大变革。自 1987 年 RISC 产品投放市场以后的短短 7 年间, 其处理器的工作速度提高了将近两个数量级, 达到了每秒执行几亿条指令。RISC 结构可使微型计算机、工作站、小型计算机、大型计算机甚至超级计算机都由同一类型的处理器组成, 这些不同机型的软件可以做到二进制兼容。RISC 的出现改变了计算机产业的产品结构。

RISC 的主要设计思想

经过多年的发展, RISC 实际上已成为一种设计思想。在 1985 年, RISC 结构的设计思想可归纳成 6

点: ①大多数指令是单周期完成的; ②采用 LOAD/STORE 结构; ③硬连线控制器; ④较少的指令数量和寻址方式; ⑤固定的指令格式; ⑥注重编译的优化。在 RISC 成为工业产品进入市场以后, RISC 思想有所发展, 因为商品化的计算机指令系统中的指令数量不能太少, 而且超大规模集成电路工艺的进展使得在单个芯片上集成几百万个晶体管是可以实现的。因此, RISC 不太强调减少指令系统的指令数量和简化硬件结构, 而是强调在流水线结构的基础上, 使执行指令的平均周期数 (CPI) 尽量减少; 或者说, 使每个周期平均执行的指令数 (IPC) 尽量提高。由于流水线 (参见计算机流水线) 指令在执行过程中, 存在着数据相关和转移相关问题, 可以用编译优化指令调度技术来提高流水线执行效率。因此, 减少 CPI 或提高 IPC 必须由硬件和软件共同来承担, 以提高系统的整体性能。

另外, 为了减少访问存储器的时间, RISC 处理机芯片内一般都包含超高速缓冲存储器 (cache), 而且数据 cache 和指令 cache 分开。

RISC 的发展趋势

RISC 的发展趋势是进一步提高指令执行的并行度。**多发射结构**可以提高并行度, 但由于在每个机器周期内发射的多条指令之间可能存在着数据相关和转移相关问题, 同时在相邻周期之间的指令也可能存在着相关, 使问题更为复杂, 从而编译优化的任务也更为艰巨。

在硬件上用寄存器旁路和内部提前方法解决数据相关问题, 相对比较容易。但有效解决转移相关仍较为困难, 可采用转移预测部件, 甚至设置专门的转移处理机。

RISC 的发展将着重于对指令流的处理和分档, 争取在发射指令以前, 依靠指令调度 (有时还加上硬件) 尽量解除指令之间的相关性, 从而提高指令执行的并行性。在流水线结构设计中, 除了传统的那些流水线功能以外, 可以有专门从事指令调度和判别功能的机制。

另外, 在分发指令以前, 就把指令流分成若干小组, 使之避免相关性, 然后发向多执行部件或多流水线部件, 这种结构称为**多线程 RISC**。

参考文献

1. 李三立, 李亚民. RISC——单发射与多发射体系结构. 北京: 清华大学出版社, 1994
2. Harold S Stone. Introduction to Computer Architecture. SRA Press, 1979 (李三立)

jingtai sui ji cunqu cunchuqi xin pian

静态随机存取存储器芯片 (static random access memory chip) 在芯片确定的地址范围内, 可以对所选择的任一地址的存储单元随机存取, 并在不断电时, 不需要周期性刷新也可稳定保持所存数据的存储器芯片。但在断电时, 静态随机存取存储器 (SRAM) 芯片所存数据会被破坏, 因此它是一种易失性存储器芯片。

分类 SRAM 芯片按接口分, 可分为同步 SRAM 和非同步 SRAM 芯片。非同步 SRAM 芯片又分为低速 SRAM 芯片 (地址访问时间 (T_{AA}) 为 120 ~ 35 ns) 和高速 SRAM 芯片 (T_{AA} 小于 30 ns)。同步 SRAM (SSRAM) 是在高速 SRAM 基础上发展起来的高速可成组访问的 SRAM 芯片。按制造工艺分类, 主要有双极型、NMOS 型、BiCMOS 或 CMOS 型。20 世纪 80 年代后, 由于 CMOS 工艺的成熟和极低的静态功耗, NMOS 和双极型 TTL 接口 SRAM 芯片已被 BiCMOS 或 CMOS 型 SRAM 芯片代替。

存储单元 在 SRAM 芯片中, 存储一位数据的基本存储单元由两个交叉耦合的反相器组成的触发器和两个由行选线控制的 MOS 管组成, 如图 1 所示。行选线选中时为高电平, 两 MOS 管通。触发器状态耦合到两条列线 BL 和 \overline{BL} , 数据可读出。写入时写入控制电路确定两列线的电平, 通过行选线选中的 MOS 管改变触发器状态, 将数据写入到存储单元。行选线未选中时为低电平, MOS 管截止, 触发器状态不能传到列线, 列线电平也不能改变触发器状态。只要不断电, 数据可永久保存, 不需周期性刷新。耦合到列线的信号幅度大, 读出延迟小, 读出时间快。但晶体管有 6 个之多, 占用的芯片面积大, 每个单元的平均价格高。

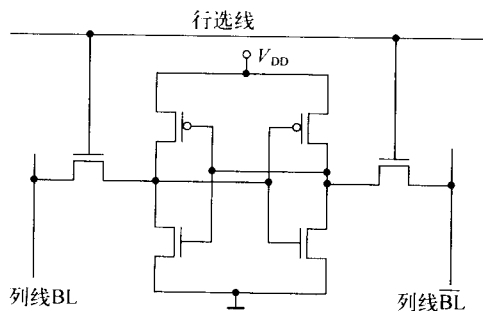


图 1 CMOS SRAM 芯片存储单元

组成和工作原理 SRAM 芯片电路由存储数据的

SRAM 存储单元阵列、控制行选择线的行地址输入缓冲、译码和行选择线驱动电路、接收列线读出数据或向列线写入数据的读出放大和写入驱动电路、控制列读写数据的列地址缓冲和译码电路、数据输入输出的接口电路和简单的数据流向控制电路组成。控制电路根据输入的写入控制 \overline{WE} 、扩展存储容量的片选信号 \overline{CS} 、高低字节控制信号 \overline{UB} 和 \overline{LB} 、控制数据输出的 \overline{OE} 信号确定芯片内部的数据通路的控制信号,以保证 SRAM 芯片正确的功能和数据流向。图 2 为 4 Mb(256 K \times 16 b) 非同步 SRAM 芯片的构成框图。16 位和 32 位数据端口的芯片要有字节控制信号。地址输入为 $A_0 \sim A_{17}$,地址空间为 256 K \times 16 b。虽然从阵列看有行地址和列地址的区分,但从外部功能上不区分。读出时,对选中芯片,只要地址变化在地址读出时间后,就可得到确切的输出数据。 \overline{WE} 为写入控制信号,通常为高电平(读操作),仅在写入时为负脉冲。必须保证在地址变化时, \overline{WE} 为高电平,只对所选的地址对应的存储单元进行写入操作。 \overline{WE} 线上绝不容许出现负毛刺干扰,以免误

写入。 $I/O_0 \sim I/O_{15}$ 为数据输入输出。 \overline{OE} 为输出控制,数据读出时为低电平。而写入时, \overline{OE} 必须为高,以保证输入数据不会和 SRAM 的读出数据冲突。只有片选信号 \overline{CS} 为低,数据才能读出或写入。读出或写入的字节由 \overline{LB} 和 \overline{UB} 控制。为低时,使对应的字节可以写入或读出。 \overline{CS} 通常由地址总线的高位地址译码产生,与 SRAM 芯片的地址输入一起限定芯片所占用的地址空间。

存储容量 芯片的存储容量专指存储单元阵列的单元总数。它决定于行地址位数 N_R 、列地址位数 N_C 和输入输出数据位数 M 。通常 M 为 4、8、16 和 32。存储容量等于 $2^N \times M$, $N = N_R + N_C$ 。存储单元阵列行选择线数为 2^{N_R} ,列线有 $2^{N_C} \times M$ 对,如图 2 所示。SRAM 芯片外部控制简单且行读出时间快,可用于小容量缓冲存储器。到 2003 年底,4 Mb(1 M \times 4 b, 512 K \times 8 b, 256 K \times 16 b) 快速 SRAM 芯片读出时间可达 8 ns(V_{CC} 为 3.3 V)。低功耗低速 SRAM 芯片的容量可达 32 Mb(2 M \times 16 b),读出时间 55~85 ns。

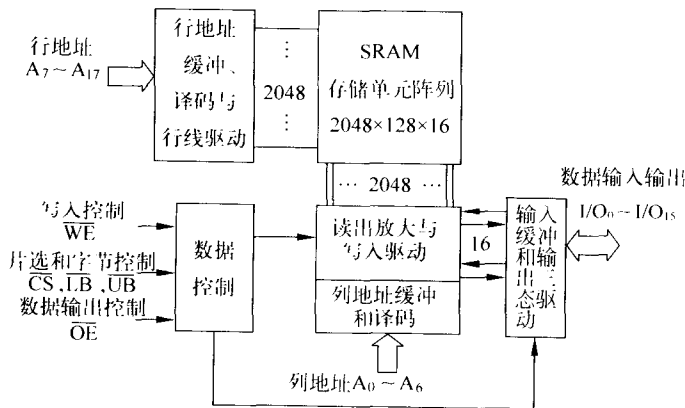


图 2 4 Mb(256 K \times 16 b) 非同步 SRAM 芯片的框图

同步 SRAM 芯片 为适应高速数字系统的要求,提高整个系统的性能,出现了同步 SRAM 芯片。它增加了同步时钟信号输入和工作方式控制输入信号。芯片的内核仍是非同步 SRAM,但输入地址、控制 and 数据的输入输出均与时钟同步,即有效的输入信号只占用一个时钟周期,减少了控制电路产生有效输入的周期数。由于同一行的存储单元在该行选中时,可同时读出到列读出放大器或由列写入数据寄存(驱动)器写入,可实现成组读写操作。即在输入第一个地址后,可读出或写入多个行地址相同,但列地址连续或等距变化的存储单元的数据。后续地

址单元的读出时间,只是输出寄存器相对于时钟信号的延迟。甚至于可实现一个时钟周期读出或写入 2 个数据(对时钟的要求会有不同),即双倍数据速率(DDR)输入输出,并且可实现流水线控制。2003 年底,1 M \times 18 b 或 512 K \times 32 b DDR SRAM 芯片的最高工作频率可达 375 MHz。1 M \times 36 b 或 2 M \times 18 b DDR SRAM 芯片相对于时钟的访问时间可达 0.45 ns。

伪静态 RAM(Pseudo SRAM)芯片 又名单晶体管 SRAM(Ut SRAM)芯片,其接口为非同步 SRAM 芯片接口,但内部为单管存储单元的 DRAM 芯片的

内核。DRAM 内核的接口信号和刷新控制在芯片内产生。这样,既利用 SRAM 芯片接口和控制简单的优点,又利用 DRAM 芯片存储单元芯片面积小,价格便宜的优点。但速度受 DRAM 内核的限制。32 Mb(2 M × 16 b) Ut SRAM 芯片的地址读出时间可达 70 ns。

SRAM 芯片的工作电压向低电压发展,以降低功耗。高速和高密度(大容量)工艺的发展允许的最高工作电压也相应降低。芯片内核和 I/O 接口的工作电压可不同。电压值可能有 5, 3.3, 2.5, 1.8V 和 1.5V。不同的芯片根据其工艺有不同的选择,以适应系统的需要。

参考文献

1. Rabaey J M. Digital Integrated Circuits: A Design Perspective. Prentice Hall, 1996. 北京:清华大学出版社,1998(影印版)

2. <http://www.samsung/Products/Semiconductor>
(孙祖希)

jingzhi tuxiang de yasuo bianma biao zhun
静止图像的压缩编码标准 (compression and coding standards of still images) 参见图像的压缩编码。

jubu sousuo suanfa

局部搜索算法 (local search algorithm) 在求解组合优化问题时,将搜索限制在局部范围的一类算法。20 世纪 60—70 年代, Lin 和 Kernighan 两人在求解旅行商问题和图划分问题时发展了经典局部搜索算法的各种技术。局部搜索算法诞生以后,在运筹学界求解优化问题的应用中一直很活跃,在算法的复杂性理论研究中也有许多成果。90 年代人工智能界的经典的约束满足问题的算法研究又掀起了局部搜索算法的研究热潮。

经典局部搜索算法包含三个步骤:

第一步:把原问题转化成一种优化形式,即在某一可行域中寻求某一目标函数的最优值的形式。例如,约束满足问题的一般形式是:给定一些变元,每个变元分别有多个可能的取值,变元的取值存在一些约束,询问满足所有约束的变元取值是否存在。在这个问题中,常把寻求满足约束条件的变元取值的布局问题,变成在所有可能取值情况组成的可行域中寻求目标函数——布局中所有不满足约束条件的冲突总数——最小值。

第二步:在可行域中定义一种邻域结构,即什

么样的两点称为相邻。比如在约束满足问题中,如果两个向量只有一个变元取值不同,则这两个向量就互称为邻点。

第三点:在邻域中反复迭代,开始局部搜索。局部搜索又包含三个步骤:首先,选择一个初始点,作为局部搜索的出发点,通常可采用随机点作为初始点。其次,确定在邻域中如何移动。可以采用见好就走的方式,即遇到比当前点好的邻点就移动。也可以采用向邻域中最好的点移动的高速下降方式。经典的局部搜索算法要求每一步移动都要更接近目标函数,因此常被称为贪心局部搜索。最后,确定如何对付局部极值点。局部极值点就是在此点的邻域中没有比它好的点。有的问题中,局部极值点一定就是全局极值点,如凸规划问题。有的问题中,局部极值点的质量足够好,已经满足需要,则也不必再寻找全局极值点了。如果局部极值点不满足要求,最简单的办法是独立重复这个下降搜索过程,即从一个新的初始点开始另一次搜索,寻找新的机会。对付局部极值点的策略有很多,这些策略又与原问题的特点相关。所以,可针对不同问题的特点发展出相应的“跳坑”策略。

从上面的介绍可以看出局部搜索算法非常简单,并且又非常有效。更有说服力的例子是旅行商问题。1990 年, Bentley 在 VAX8550 机器求解 10^6 个城市的随机实例, 3.8 小时内得到的解仅比最优解相差不到 3.5%。

实际上,局部搜索算法还代表了相当一大类问题的现有算法的共同特征。常见的一些具有多项式算法的问题,如最小生成树、最大流、最大匹配,还有排序问题,算法上均是一种局部搜索的思想,而且具有局部最优就是全局最优的好性质。连续优化问题由于不能像组合优化问题那样至少可以通过枚举来求解,因此所有算法均是局部搜索性质的算法。近年来非常流行的计算智能中的一些方法,如模拟退火算法、遗传算法等都可看作是在经典局部搜索算法基础上的一种变形。(卜东波)

juyuwang

局域网 (local network) 将小区域范围内的数据通信设备互连在一起以高的数据传输速率互相通信的一种通信网络。所谓小区域可以是 1 个建筑物、1 个校园或者是一个小于 100 km 的区域。数据通信设备则包括计算机、终端和各种外围设备等。数据通信设备在网络中有时被称为站或站点。起初的局域网数据传输速率均小于或等于 100 Mb/s。进

入 20 世纪 90 年代以后,数据传输速率超过 100 Mb/s 的局域网相继推出,如千兆网的数据传输速率达到 1 000 Mb/s,通常称之为**高速局域网**,而把数据传输速率小于 100 Mb/s 的局域网称为**低速局域网**。上述定义的局域网是一个通信网,只有加上高层协议和网络软件才能组成计算机局域网。局域网的典型特征是:①传输速率高(1~1 000 Mb/s);②地理覆盖范围小(小于 100 km);③传输误码率低($10^{-8} \sim 10^{-11}$)。

按介质的通信方式来区分,局域网可分为基带和宽带两类。基带局域网在传输介质上传输的是单一传输速率的数字信号,而宽带局域网在传输介质上传输的是经高频调制的模拟信号。这是一种单向传输技术,同一介质可传输多个不同的频道,以支持数据通信、电视和无线电信号。

常用的局域网有以太网、权标环网和权标总线网以及光纤分布式数据接口、快速以太网、千兆位以太网和异步传送模式局域网等。常用局域网的主要技术特征如表 1 所示。

表 1 常用局域网的主要技术特征

拓扑结构	总线、环状、树状、星状
传输介质	同轴电缆、双绞线电缆、光缆、无线电波
数据传输速率(Mb/s)	4、10、16、100、1 000
介质访问控制方法	CSMA/CD、权标传递、CSMA/CA
介质传输技术	基带、宽带、IR、FHSS、DSSS

局域网拓扑结构 局域网拓扑结构是指网络站点及其连接的几何布局。构成局域网的拓扑结构有多种,包括以太网所用的总线、树状或星状拓扑结构,权标环网所用的环状拓扑结构等。

局域网介质访问控制方法 美国电气与电子工程师学会(IEEE)下属的 802 委员会对局域网制定了许多协议标准,统称 IEEE 802 标准。它提出的**局域网基准(参考)模型**包括了**开放系统互连基准(参考)模型**的最低两层(物理层和数据链路层)的功能,也包括网间互连的高层功能和管理功能。其中,数据链路层又划分为逻辑链路控制子层和介质访问控制子层(参见**局域网逻辑链路控制子层**和**局域网介质访问控制子层**)。局域网介质访问控制方法是局域网介质访问控制子层的主要功能,即对物理传输介质进行访问管理。起初的局域网都是采用共享介质的通信技术,常见的有带碰撞检测的载波侦听多址访问—冲突检测(CSMA/CD)(参见**以太网**)、权标传递(参见**权标环网**)等方法。共享介质通信

技术要求发送站点先向连接介质上的所有站点广播其要求或标记,接着要确认是否被获准发送,然后再发送信息。可减少在介质上的碰撞损失。但由于这种方法存在不足之处,故已出现了一些改进和增强的办法。尤其是采用**网桥或交换机**来互联 LAN 的技术已被广泛使用。用网桥或交换机互联起来的一组 LAN 站点统称为**桥接局域网**。图 1 为 1 个桥接局域网的例子,这里 A 和 B 是网桥,C、D 和 E 为交换机。由 A 和 B 把两个以太网和**高速光纤环网(FDDI)**相连是使用的共享介质通信技术,而把快速以太网和 FDDI 网相连是通过交换机 C、D 和 E 进行的,其中 C 为主干交换机。

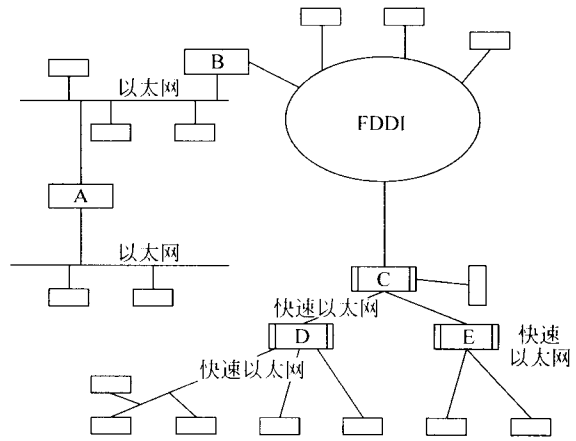


图 1 桥接局域网

传输介质 局域网的传输介质可分为有线和无线两类。常见的有线类传输介质有**同轴电缆、双绞线电缆和光缆**。无线类传输介质有无线电波和光波。无线电波是近来新兴的**无线局域网**用的介质。

对于双绞线,由于局域网的传输速率已经达到 10 Mb/s 以上,甚至 100 Mb/s,因此,必须选用数据级双绞线。以无屏蔽双绞线(UTP)为例,选择类型 3、4 以及 5,这 3 种类型可分别适应以太网的 10 Mb/s,权标环网的 16 Mb/s 和光纤分布式数据接口或快速以太网的 100 Mb/s 3 种使用环境。同轴电缆常用于以太网,它的频宽与抗外界电磁场干扰能力均比 UTP 强。在基带或宽带以太网上分别选择 50 Ω 或 75 Ω 的同轴电缆作为传输介质。光缆作为局域网传输媒体可获得很高的传输速率,很强的抗外界电磁场干扰能力,极小的电磁辐射以及较长的网络段跨距等。由于光缆内的光信号有单向性和不宜分流的特点,因此以光缆作为传输介质的局域网其拓扑结构常选择星状、簇状或环状。

参考文献

1. Alan chambers. IEEE Standards for Local and Metropolitan Area Network. Overview and Architecture, 1999. 3. 16

2. 胡道元. 计算机局域网(第三版). 北京: 清华大学出版社, 2002 (黄令恭 张公忠)

juyuwang jizhun (cankao) moxing

局域网基准(参考)模型(local area network reference model)

基于国际标准组织(ISO)的开放系统互连基准(参考)模型又适用于局域网环境的模型。局域网基准(参考)模型包含 OSI 基准(参考)模型的物理层,数据链路层和其他更高层的部分内容。美国电气与电子工程师学会(IEEE)标准委员会制定的局域网基准(参考)模型(IEEE 802 基准(参考)模型)如图 1 所示。物理层与开放系统互连基准(参考)模型(OSI/BRM)的物理层相同,而开放系统互连基准(参考)模型的数据链路层被分为逻辑链路控制(LLC)和介质访问控制(MAC)两个子层,这样处理的目的是要把数据链路层中涉及介质的部分和与介质无关的部分分开。

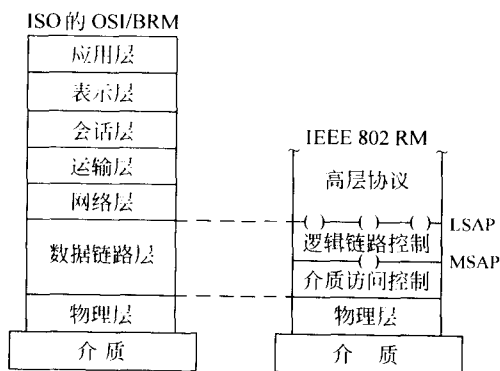


图 1 IEEE 802 基准(参考)模型

逻辑链路控制(LLC)子层(参见局域网逻辑链路控制子层)向高层提供一个或多个链路服务访问点(LSAP)接口,完成帧的收、发功能。

介质访问控制(MAC)子层(参见局域网介质访问控制子层)的主要功能是对数据传输介质进行访问管理,对帧进行定界和识别,对目的站点寻址,对 LLC 的协议数据单元(PDU)进行透明传送和差错检测等(参见网络数据单元)。

物理层提供在物理层实体间发送和接收比特的能力。它提供给 MAC 子层的服务是比特的传送,如起初的以太网、令牌环网的 MAC 帧在物理介质上实

际传输的就是一个接一个的比特。但局域网发展至今,尤其当数据传输速率大于 100 Mb/s 时,物理层常把一组比特(如 4、5 或 8 个比特)表示为不同的多元符号(有时称为码组),并以它为单位来进行传输。

为了指导某个具体标准的开发,IEEE 标准委员会在局域网基准(参考)模型基础上又采用了一种实施模型(IEEE 802 实施模型)。局域网实施模型更重视物理层标准的复杂性。如对于某种介质访问控制机制来讲,有可能要在多种不同的物理介质(如无屏蔽双绞线电缆,屏蔽双绞线电缆,光纤等)中选一种运行。这样,物理层规程中有一部分是与介质无关的,如处理信号编码、同步化等问题的规程;另有一部分是与介质相关的,如针对特定介质的电气和机械方面问题的规程。从实施角度看,这两部分是可以分别处理的。

1980 年 2 月,IEEE 的计算机学会举行了第一次“LAN 标准委员会”会议(故称 802 项目)。经过多年的实践,它已发展成为目前 IEEE 的局域网和城域网标准委员会(LMSC),仍可叫 802 项目。它下设 13 个工作组,2 个技术顾问组和 1 个执行委员会。各工作组按照上述 LAN 模型制定相应的标准(参见局域网协议标准)。

参考文献

Alan chambers. IEEE Standards for Local and Metropolitan Area Network. Overview and Architecture, 1999. 3. 16, < <http://www.ieee.org/1/pages/802.html> >

(黄令恭 张学尧)

juyuwang jiezhi fangwen kongzhi fangfa

局域网介质访问控制方法(medium access control method of local area network)

在局域网中对数据传输介质进行访问管理的方法。起初采用共享介质方式的带碰撞检测的载波侦听多址访问(CSMA/CD)、令牌传递或光纤分布式数据接口(FDDI)等方法。但随着局域网应用的不断扩展,这种共享介质方式对任何端口上的数据帧都不加区别地进行传送时,经常会引起网络冲突,甚至阻塞,所以目前经常采用利用网桥、交换机等方法将网络分段,以减少甚至消除网络冲突的交换方式。

共享介质方式中最常用的介质访问控制方法为带碰撞检测的载波侦听多址访问(CSMA/CD)和令牌传递。

CSMA/CD 是以太网中采用的介质访问控制方法。它的原理是连接在以太网总线上的任何设备在

任何时候都可以去尝试发送一个帧,如果此时总线空闲,总线就开始传送这个帧。如果此时另一个设备也在尝试发送,就要发生碰撞。碰撞双方就会放弃这次尝试。各方随机等待一段时间后,再去试送,直到发送成功为止。

权标传递是**权标环网**中采用的介质访问控制方法。**权标**是一个专用的控制帧。它不停地在环路上各站点间传递着。它被用来标志环路是否空闲,即站点是否可以发送数据帧。若某个站点有数据要发送,它就在环路上等待权标帧的到来,一旦权标帧到来,即占用它,然后发送数据帧。当数据帧发送结束时,再将权标帧释放,让其在环路上传递下去,以便环路上其他站点发送数据帧。

交换方式是不同于共享介质方式的另一种局域网的介质访问控制方式。它是为了解决网络冲突,进一步提高网络有效带宽的一种介质访问控制方式。对一个使用共享介质方式的有 30 个站点的传输速率为 10 Mb/s 以太网来说,其平均(每站)单位时间吞吐量为 1/3 Mb/s。但随着用户数增多,冲突频繁发生,引起网络阻塞,则其单位时间吞吐量要随着带宽利用率的变化而变化。尤其当带宽利用率超过 37% 后,单位时间吞吐量将急剧下降,仅有额定的 30% 可被使用,这样平均(每站)单位时间吞吐量仅为 100 kb/s。解决这个问题的一种方法就是将一个局域网分段。减少每段的站点数。站点越少,每个站点享用的带宽就越多。如每个网段只有一个站点,它就可以享用全部带宽。

交换机在**局域网**中处在相当于**集线器**的位置,但不像集线器那样要向所有端口重发输入帧,而是去观察此帧的目的地址和源地址,以确定将此帧转发到哪个输出端口上去。一个交换机通常由 I/O 缓冲器、I/O 端口和交换部件三部分所组成,经常采用的是“穿通”和“存储转发”两种内部转发技术。

在采用**穿通技术**时,交换机只要读入目的地址,确定了输出端口后,就开始转发此帧到输出端口去。它的优点是具有较低的交换延迟,而缺点是不管此帧有无差错均转发。而在用**存储转发技术**时,交换机在接收并分析了整个帧后,才确定是否向输出端口转发此帧。这里包括要进行**循环冗余检错**等操作,显然要多费时间。

参考文献

Christensen K J. Local Area network—Evolving from Shared to Stitched Access. IBM Systems Journal, 1995, 34(3)

(黄令恭)

juyuwang jiezhil fangwen kongzhi ziceng
局域网介质访问控制子层 (medium access control sublayer of local area network) **局域网基准(参考)模型**中的一个子层。介质访问控制(MAC)子层通过一个服务访问点(MSAP)向**逻辑链路控制(LLC)子层**(参见**局域网逻辑链路控制子层**)提供服务,完成数据链路层中的成帧、拆帧、流控,以及对**局域网**数据传输介质的正确使用等功能。介质访问控制(MAC)是指**局域网(LAN)**各站点对共享**传输介质**可使用的权力和应遵守的规则。

MAC 子层的主要功能是:①帧的定界和识别;②目的站点的寻址;③源站点寻址信息的传送;④逻辑链路控制子层的协议数据单元(PDU)的透明传送;⑤差错检测;⑥对物理传输介质的访问管理。

对应不同的**局域网网络拓扑结构**和传输介质,局域网介质访问控制采用不同的方法。常见的有:①用于总线或树状结构,采用**双绞线电缆**或**光纤**为介质的 CSMA/CD 方法;②用于环状结构,采用双绞线或**光纤**的**权标传递**方法;以及③用于**无线局域网**的 CSMA/CA 方法。

当 LAN 上用户站点数增多时,其业务量也随之增加,可能引起通信性能的下降,这是共享介质访问的 LAN 共同存在的问题。解决的方法是用**网桥**将之分段,以减少同一个**局域网**上的用户站点数和业务量。最常用的一种网桥是透明网桥,另外一种源路由网桥主要用于**权标环网**上(参见**网桥**)。

除了网桥,在 MAC 子层工作的常用网络设备还有网络交换机,这里指的是帧交换机和 ATM 交换机,如快速以太网交换机,千兆位以太网交换机和高速**权标环交换机**等。

参考文献

Stallings W. Local and Metropolitan Area Networks, 6/e. Prentice Hall, Inc., 2004 (黄令恭)

juyuwang luoji lianlu kongzhi ziceng
局域网逻辑链路控制子层 (logical link control sublayer of local area network) **局域网基准(参考)模型**中的一个子层,它通过一个或多个服务访问点(LSAP)向高层提供无连接的和面向连接的数据传送服务。逻辑链路控制(LLC)子层利用各种介质访问控制(MAC)子层(参见**局域网介质访问控制子层**)去适应不同类型**局域网**的工作,然后

再重新组合它们,以便统一向高层提供服务。

局域网协议标准 (IEEE 802) 定义了三种逻辑链路控制服务类型:

(1) 类型 1 不确认的无连接服务。信息帧在 LLC 实体间交换,无须同等层实体间事先建立逻辑链路。对这种信息帧既不确认,也无任何流量控制或差错恢复功能。

(2) 类型 2 面向连接服务。任何信息帧在交换前,在一对 LLC 实体间必须建立逻辑链路。在数据传送时,信息帧依次发送,并提供差错恢复和流量控制。

(3) 类型 3 确认的无连接服务。无须事先建立连接,但信息帧的发送和接收必须经过确认。这种服务用于传送某些非常重要且时间性很强的信息。

LLC 子层把发往(或来自)高层的分组封装成 LLC 帧,即把分组作为 LLC 帧的数据字段,加上目的服务访问点(DSAP)字段和源服务访问点(SSAP)字段以及一个控制字段构成一个 LLC 帧(图 1)。这个 LLC 帧还要有来自(或发往)介质访问控制(MAC)子层,作为 MAC 帧中的数据字段。LLC 帧中控制字段的格式是仿效高级数据链路控制规程(HDLC)中的平衡模式制定的。如服务类型 2 的 LLC 帧可用其控制字段把 LLC 帧分为信息帧、监控帧和无编号帧,其中信息帧用来传送数据信息;监控帧用于确认、流量控制和差错控制;无编号帧用于连接建立、连接终止及其他控制功能。

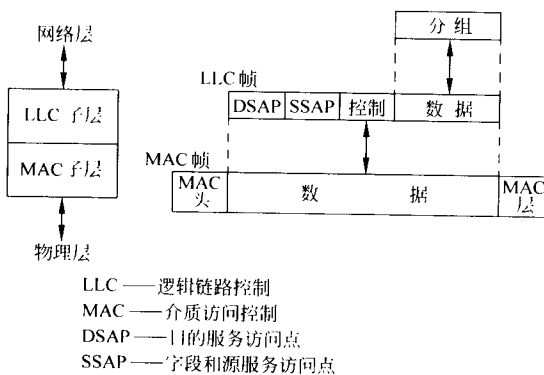


图 1 LLC 帧

随着局域网 (LAN) 应用的日益普遍, LAN 中的管理和安全已列入局域网协议标准 (IEEE 802)。逻辑链路控制子层中的“LAN 管理”模块提供在 LAN

站点间交换管理信息用的数据链路层管理协议,并为所有局域网协议标准规定了被管理的对象。逻辑链路控制子层中的安全数据交换 (SDE) 实体,直接在介质访问控制子层上面提供安全的无连接服务。

参考文献

1. Stallings W. Local and Metropolitan Area Networks, 6/e. Prentice Hall, Inc., 2000
2. 张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999 (黄令恭)

juyuwang tuopu jiegou

局域网拓扑结构 (topology of local area network)

在局域网中,网络站点(计算机或其他设备)及通信线路的几何布局,即网络中结点及其通信链路的互连模式。局域网的拓扑结构有:全互连结构、总线结构、星状结构、环状结构、树状结构和簇状结构等,但以总线、星状和环状结构最为常见。

全互连结构 如果一个网络只连接几台设备,最简单的方法是将它们都直接相互连在一起,这种连接称为点对点连接。用这种方式形成的网络称为全互连网络,如图 1 所示。这种方式只有在涉及地理范围不大,设备数很少的情况下才有使用的可能。

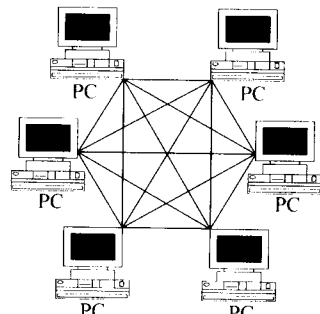


图 1 全互连结构

总线结构 使用同一介质连接所有站点的一种方式,即连接站点的物理介质由所有设备共享,如图 2 所示。任何一个站点发送的信号都可以沿着介质传播,而且能被其他所有站点接收。总线拓扑的优点是:费用低,电缆长度短,易于布线和维护;结构简单,传输介质又是无源元件,从硬件的角度看十分可靠;站点扩充容易,某个端用户失效或增、删不影响其他站点或端用户通信。缺点是没有对网络进行集中控制,故障检测需要在网上的各个站点上进

行;一次仅能一个端用户发送数据;网络覆盖范围受到限制等。

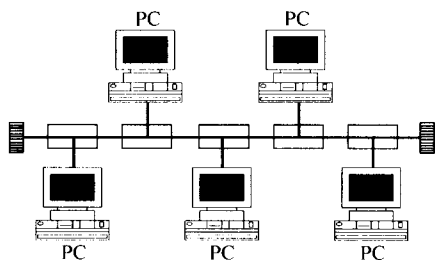


图2 总线结构

星状结构 由通过点到点链路接到中央结点的各站点组成。星状网络中有一个惟一的转发结点,每一台计算机都通过单独的通信线路连接到中央结点,如图3所示。其中处于中心位置的网络设备称为**集线器(Hub)**或**交换机**。其优点是:利用中央结点可方便地提供服务和重新配置网络;单个连接点的故障不会影响全网,便于维护;访问协议十分简单。缺点是每个站点直接与中央结点相连,需要大量电缆,因此费用较高;一旦中央结点产生故障,则整个系统趋于瘫痪,所以对中央结点的可靠性和冗余度要求很高,通常采用双机热备份。

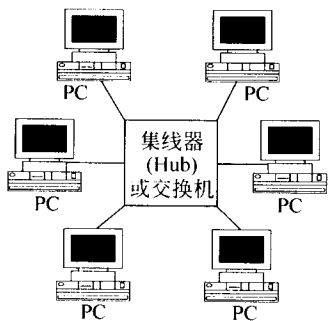


图3 星状结构

环状结构 由连接成封闭回路的网络结点组成,如图4所示。在环状网络中信息流一般是单向的。每个结点都向它的下游结点转发它收到的信息包。信息包在环网中环绕一圈,最后由发送结点回收。由于多个结点共享一个环,就需要一种介质访问控制方法来决定每个结点何时发送数据。它的优点是它提供的对介质访问的灵活控制;消除了端用户通信时对中心系统的依赖性。其缺点是如果环的某一点断开,环上所有结点间的通信便会终止等。

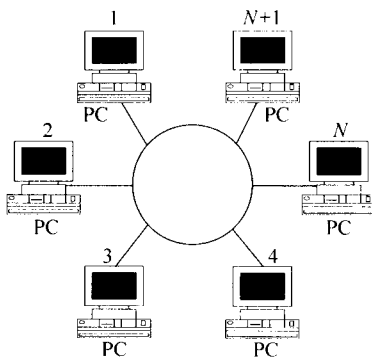


图4 环状结构

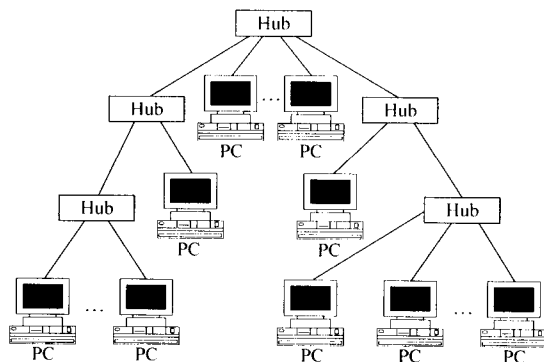


图5 树状结构

为了扬长避短,使网络易于扩展,把故障隔离在网络的某一局部,可以将上述结构进行组合和改进,从而形成了树状结构、簇状结构和星状环结构等。例如,图5就是树状结构,它是对总线结构和星状结构的组合和改进。

参考文献

肖代华. 几种常见的局域网拓扑结构. 电脑报, 1998(11) (张博锋)

juyuwang wuli jiezhì wuguan ziceng
局域网物理介质无关子层 (physical medium-independent sublayer of local area network)

局域网基准(参考)模型物理层中的一个子层。它处理信号的编码与译码、前导码的生成或取消、传输比特和接收比特等与传输介质不直接相关的事件。局域网的物理层还要去处理与传输介质密切相关的事件(参见**局域网物理相关子层**),才能完成其向介质访问控制子层提供“比特的传送”服务。

在基带局域网中,物理层需提供收发未经调制信号(数字信号)的功能。在宽带局域网或**无线局域网**中,物理层需提供收发指派在专用频道上调制信号(模拟信号)的能力。参见**局域网**。

物理层在传输介质上运送原始比特,并非只能一个比特一个比特地进行传输,很多场合可采用多个比特来组成一个**码组**,以码组为单位来进行传输。如在**光纤分布式数字接口(FDDI)**、100 BASE X 等标准中采用的 4B/5B 码,就是以 5 个比特为一个码组,它可代表 4 个比特的 16 个二进制数据以及其他一些控制标记。常有的码组还有 5 B/6B 码、8 B/10B 码和 8 B/6T 码等。

物理层常用到的网络设备是**中继器**和**集线器**。在**权标环网**中的环路就是由每个站点的中继器连成的;在**以太网**中采用中继器把各个网段连起来。通过中继器对所传输的数据信号整形放大再送出去,这样做显然可以扩大局域网的地理覆盖范围,但所覆盖的范围仍然属于一个单独的**访问域**。在一个访问域中的各站点用共享介质方式通信,即在一个给定的时间内至多只有一个站能进行发送,其他均为接收站。如两个站同时进行发送,就会引起冲突。

集线器早已用在星形拓扑的以太网,其作用类似中继器。当一个站要进行发送时,集线器同样会中继这些信息,并在连向其他各个站点的输出线上将它们发送出去。虽然用在星形拓扑,但逻辑上还是属于一个单独的访问域。近年来随着网络管理的需要,已推出了“智能”集线器,它可通过软件命令方式将各个端口置为不同的网段,以减少各网段内的冲突,提高网络的总带宽。有些智能集线器还支持**简单网络管理协议(SNMP)**。

参考文献

Stallings W. Local and Metropolitan Area Networks, 6/e. Prentice Hall Inc., 2000 (黄令恭)

juyuwang wuli jiezhi xiangguan ziceng
局域网物理介质相关子层(Physical medium-dependent sublayer of local area network, PMD) **局域网基准(参考)模型**物理层中的一个子层。它处理通过**光纤**、**同轴电缆**、**屏蔽双绞线**以及**无线电波**等**传输介质**进行信息收、发与**传输介质**密切相关的事件。

在局域网实施模型(参见**局域网基准(参考)模**

型)中已将物理层分为与介质相关和无关两部分,这两部分经过一个接口相连接。最早将物理介质相关(PMD)作为物理层中的一个子层并在协议标准中提出出来的是**光纤分布数据接口(FDDI)**。在其 1985 年规程草案中已明确 PMD 要求规定光纤的驱动器和接收器、介质相关的编码要求、连接器、光旁路设施以及有关硬件的物理特征。在 20 世纪 90 年代的快速以太网中也借用了 FDDI 的 PMD 机制。

千兆位以太网的基准(参考)模型(参见**千兆位以太网**)中已将 PMD 子层明确列入。相对于不同的传输介质(多模光纤、单模光纤、小于 25m 的短距离专用屏蔽电缆以及小于 100m 的 4 对 5 类无屏蔽双绞线),该参考模型提供了不同的 PMD 机制(支持建筑物间互连的 1 000 BASE-LX PMD,支持建筑物内水平布线的 1 000 BASE-SX PMD,支持设备集群间互连的 1 000 BASE-CX PMD 以及 1 000 BASE-T PMD)。

随着无线局域网逐步接入应用,相应的窄带、扩频和红外三类技术也将列入 PMD 子层的范围(参见**无线局域网**)。

参考文献

Stallings W. Local and Metropolitan Area Networks, 6/e. Prentice Hall Inc., 2000 (黄令恭)

juyuwang xieyi biao zhun

局域网协议标准(Protocol Standards of Local Area Network—IEEE 802 Std) 基于开放系统互连基准(参考)模型的适用于**局域网**环境的协议标准。它是美国电气和电子工程师学会(IEEE)802 课题组制定的局域网协议标准。

局域网(LAN)采用共享的传输介质,使其能在一个短暂时间间隔内从多个信源将信息快速地传送到多个信宿,出现“多对多”的信息交换情况。**局域网基准(参考)模型**只包含**开放系统互连基准(参考)模型**的最低两层,即物理层和数据链路层。由于局域网中的介质访问控制(MAC)比较复杂,开放系统互连基准(参考)模型的数据链路层又细分为逻辑链路控制(LLC)和介质访问控制(MAC)两个子层。LLC 子层提供一个或多个服务访问点(SAP)和高层联系,以复用的形式建立“多点—多点”之间的数据通信;MAC 子层用来管理经过链路进行的多路通信以及普通数据链路层中所具有的成帧、拆帧、流控等功能,即具体管理通信实体接入信道而建立数据链路的控制过程。

1980 年以来,许多国家和国际标准化组织都在进行局域网的标准化工作。美国电气和电子工程师学会(IEEE)802 课题组制定了许多有关局域网协议的标准,其中有些标准已被国际标准化组织(ISO)采纳作为 ISO 的国际标准发布,称为 ISO 8002-X 标准,或称 IEEE 802-X 标准。除了 IEEE 802 所开发的局域网标准外,还有其他一些标准化组织也开发

了一些局域网标准,如美国国家标准学会(ANSI)X3T9.5 委员会开发的光纤分布式数据接口(FDDI)标准。

IEEE 802 是一个标准系列,其综合情况见图 1 (包括个别有关 ISO 标准),图中各协议的含义及内容简释如下:

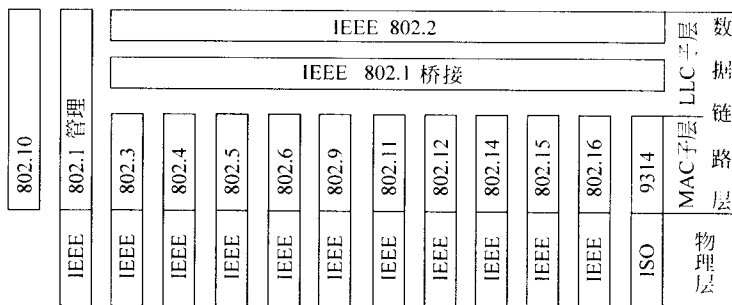


图 1 局域网标准一览

(1) **IEEE 802 LAN 标准**: 局域网概述和体系结构,1997 年发布。

802.1B LAN 和城域网(MAN)管理,1995 年发布。定义了一种进行远程管理用的,与 OSI 管理一致的体系结构和环境。

802.1D MAC 桥接,1998 年发布。提供快速业务流通功能和过滤业务。

802.1E 系统负载协议,1994 年发布。规定用于管理有关在 802LAN 上装载的一组服务和协议。

802.1F 用于 IEEE 802 管理信息的公共定义和过程,1993 年发布。

802.1G 远程 MAC 桥接,1998 年发布。规定了本地 MAC 网桥操作远程网桥的方法。

802.1H 在局域网中以太网 2.0 版 MAC 桥接,1997 年发布。

802.1Q 虚拟桥接局域网,1998 年发布。

(2) **IEEE 802.2 逻辑链路控制(LLC,包括简单无连接、连接方式,带确定无连接等服务)**,1998 年发布。

(3) **IEEE 802.3 带碰撞检测的载波侦听多址访问(CSMA/CD)的访问方法和物理层规范**,1998 年发布。

802.3ac 虚拟局域网(VLAN)的帧扩展,1998 年发布。

802.3ab 1000 BASE-T 物理层参数和规范,1999 年发布。

802.3ad 多重链接分段的聚合协议,2000 年发布。

(4) **IEEE 802.4 逻辑令牌总线访问方法和物理层规范**,1990 年发布。1997 年附加了 802.4h 协议,它是对单通道总线物理层实体 BNC 连接器和曼彻斯特编码信号方法的选择使用做出的规范。

(5) **IEEE 802.5 令牌环访问方法和物理层规范**,1997 年发布。

802.5r 专用的令牌环的运行,1997 年发布。

802.5t 100 Mb/s 高速令牌环访问方法,2000 年发布。

(6) **IEEE 802.6 城域网(MAN)访问方法和物理层规范**,1994 年发布。1995 年附加了 MAN 的分布式队列双总线(DQDB)子网上面向连接的服务协议。

(7) **IEEE 802.9 在 MAC 和物理层上综合话音和数据(IVD)局域网技术**,1996 年发布。

(8) **IEEE 802.10 可互操作的局域网安全标准(SILS)**,1998 年发布。还附加了安全体系结构框架的 802.10a 和密钥管理的 802.10c。

(9) **IEEE 802.11 无线局域网的 MAC 协议的物理层规范**,1999 年发布。还附加了 5GHz 波段

高速物理层的 802.11a 和对 2.4GHz 高速物理层扩充的 802.11b。

(10) **IEEE 802.12** 需求优先协议, 1998 年公布了 100 Mb/s 需求优先访问方法以及物理层和中继器规范, 1998 年还附加了全双工操作规范。

(11) **IEEE 802.14** 用于有线电视 (CATV) 宽带通信的标准, 1998 年发布。

(12) **IEEE 802.15** 无线私人网 (WPAN), 是一种开放性短距离无线通信技术, 又称蓝牙技术。

(13) **IEEE 802.16** 宽带无线访问标准, 由两部分组成:

802.16.1 固定宽带无线访问的无线界面;

802.16.2 宽带无线访问系统的共存。

(14) **ISO 9314** 光纤分布式数据接口 (FDDI), 1989 年发布。

随着局域网应用的逐渐深化, 将会有新的协议标准出现。

参考文献

1. IEEE 802 Working Group Home Page. Howard Franzier, Aug. 24, 1999, <http://grouper.ieee.org/groups/802/dots.html>.

2. 802 overview & architecture. Alan Chambers, Draft 27, Mar. 16, 1999, <http://www.ieee802.org/1/page/802.html>.

3. 胡道元. 计算机局域网 (第三版). 北京: 清华大学出版社, 2002 (丁友东)

juzhen jisuan

矩阵计算 (matrix computation) 矩阵的代数运算、乘积快速计算和快速求逆等的总称。 $m \times n$ 个数 a_{ij} ($i = 1, 2, \dots, m; j = 1, 2, \dots, n$) 的矩形阵

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad (1)$$

称为 m 行 n 列矩阵或 $m \times n$ 阵, 简记为 $A = (a_{ij})_{m \times n}$, a_{ij} 称为 A 的元素。如果 $m = 1$, 称 A 为行矩阵; $n = 1$, 称 A 为列矩阵; 如果 $m = n$, 称 A 为方阵, n 称为 A 的阶数。当 $i \neq j$ 时, $a_{ij} = 0$, 称 A 为对角矩阵。当对角矩阵对角线上的元素全为 1 时, 称为单位矩阵, 常用 I (或 E) 表示。

矩阵的相等、加、减、乘、转置与共轭 两个阶数相等的矩阵 $A = (a_{ij})_{m \times n}$, $B = (b_{ij})_{m \times n}$ 。 $A = B$ 是指

它们的对应元素相等 ($a_{ij} = b_{ij}$); $C = A + B$ 是指 A 和 B 对应元素相加 ($c_{ij} = a_{ij} + b_{ij}$) 所形成的矩阵。矩阵加法满足结合律和交换律。矩阵 $-A = (-a_{ij})_{m \times n}$ 称为 A 的负矩阵, $A + (-A) = O$ 。两个 $m \times n$ 矩阵的减法是 $A - B = A + (-B)$ 。

如果 A, B 分别是 $m \times n$ 与 $n \times q$ 矩阵, 称 B 对于 A 是可乘的, 其乘积 $C = AB$ 是一个 $m \times q$ 矩阵, 它在

(i, j) 位置上的元素为 $c_{ij} = \sum_{k=1}^n a_{ik} b_{kj}$ ($i = 1, 2, \dots, m; j = 1, 2, \dots, q$)。假如 A, B 是同阶方阵, 那么 AB 与

BA 都有定义, 与数的情况不同, 这里可能出现 $AB \neq BA$ 以及当 $A \neq O$ 及 $B \neq O$ 时, $AB = O$ 的情况。矩阵乘法满足结合律和分配律。

$m \times n$ 矩阵 A 的转置矩阵是一个 $n \times m$ 矩阵, 记为 A^T 或 A' 。对所有 i, j , A^T 的第 i 行是 A 的第 i 列, A^T 的第 j 列是 A 的第 j 行。如果 B 对于 A 是可乘的, 那么有 $(AB)^T = B^T A^T$ 。如果 $A^T = A$, 称 A 是对称矩阵; 如果 $A^T = -A$, 称 A 为反对称矩阵。显然, 对称矩阵与反对称矩阵一定是方阵。如果有 $A^T A = AA^T = I$, 称 A 是正交矩阵。矩阵 A 的共轭矩阵 \bar{A} 是 $\bar{A} = (\bar{a}_{ij})_{m \times n}$, 这里 \bar{a}_{ij} 是 a_{ij} 的共轭复数。如果 $\bar{A} = A^T$, 则称 A 为厄米特矩阵。

快速矩阵乘法 设 A, B 是 n 阶方阵, 利用矩阵乘法定义, 计算 $C = AB$ 的运算量是 $O(n^3)$ 。1969 年, V. Strassen 提出了运算量为 $O(n^{2.8074})$ 的快速矩阵乘法。该法的基本思想是将 2 阶矩阵乘法所需的乘法数从 8 个减为 7 个, 对于阶为 2 的乘幂的矩阵分解为 2×2 的矩阵块, 块中元素均为 $n/2$ 阶方阵, 一直递归分解下去。经 $\log_2 n - 1$ 步后化为 2 阶矩阵。具体作法如下: 设 $n = 2^t$, 并记

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \quad (2)$$

要计算

$$C = AB = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

按矩阵乘法定义需 8 次矩阵块乘法与 4 次矩阵块加法。现在先按下列式子 (每个式子一个乘法) 算出 M_i ($i = 1, 2, \dots, 7$), 即

$$\begin{aligned} M_1 &= (A_{11} + A_{22})(B_{11} + B_{12}), & M_2 &= (A_{21} + A_{22})B_{11} \\ M_3 &= A_{11}(B_{12} - B_{22}), & M_4 &= A_{22}(B_{21} - B_{11}) \\ M_5 &= (A_{11} + A_{12})B_{22}, & M_6 &= (A_{21} - A_{11})(B_{11} + B_{12}) \\ M_7 &= (A_{12} - A_{22})(B_{21} + B_{22}) \end{aligned}$$

(3)

再按下列各式组成乘积 $C=AB$ 的元素 (只用加减法)

$$\begin{aligned} C_{11} &= M_1 + M_4 - M_5 + M_7, & C_{12} &= M_3 + M_5 \\ C_{21} &= M_2 + M_4, & C_{22} &= M_1 + M_3 - M_2 + M_6 \end{aligned} \quad (4)$$

对于式 (3) 中的每一个 $n/2$ 阶矩阵乘法, 又可重复利用式 (2) 至式 (4), 如此递归作下去, 可知 V. Strassen 提出的矩阵乘法的总的运算量为 $O(n^{2.8074})$ 。若矩阵阶数 n 不是 2 的乘幂, 取 t 满足 $2^{t-1} < n < 2^t$, 将 A, B 扩充成 2^t 阶矩阵, 对扩充后的矩阵应用 V. Strassen 提出的矩阵乘法, 然后可求出 AB , 所需运算量仍为 $O(n^{2.8074})$ 。此外还有 A. Waksman 提出的快速矩阵乘法法和 V. Pan 提出的快速矩阵乘法等。

行列式 设矩阵 $A = [a_{ij}]$ 是 n 阶方阵, 其行列式定义为一个数, 用

$$|A| = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix}$$

或 $\det A$ 表示, 它是 A 的所有元素的函数, 其值是所有符合以下条件项的代数和。

(1) 每项是 A 中每行、每列各取一个元素组成的 n 个元素的乘积, 可记为

$$a_{1j_1} a_{2j_2} \cdots a_{nj_n}$$

其中, j_1, j_2, \dots, j_n 是 $1, 2, \dots, n$ 的一个排列。

(2) 每项 $a_{1j_1} a_{2j_2} \cdots a_{nj_n}$ 前带正负号, 以 $1, 2, \dots, n$ 的顺序为标准来比较排列 $(j_1 j_2 \cdots j_n)$ 的逆序数是奇次或偶次而决定: 奇带负号, 偶带正号。

行列式的计算常用高斯消去法来完成。令

$$a_{ij}^{(1)} = a_{ij} \quad (i, j = 1, 2, \dots, n)$$

则逐次消去过程为

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - a_{ik}^{(k)} a_{kj}^{(k)} / a_{kk}^{(k)}$$

$$(k = 1, 2, \dots, n-1; \quad i, j = k+1, k+2, \dots, n)$$

此时

$$|A| = \prod_{k=1}^n a_{kk}^{(k)}$$

在消去过程中要用 $a_{kk}^{(k)}$ 作除数, 故要求 $a_{kk}^{(k)} \neq 0$ 。同时为了减少误差的积累, 应取大元素作除数, 这可用选主元的办法解决。例如采用行主元格式, 即取 $a_{il}^{(k)} = \max_{j \geq k} |a_{ij}^{(k)}|$, 如 $l \neq k$, 则将第 k 列和第 l 列对调, 同时将行列式变号。如 $a_{il}^{(k)} = 0$, 则 $|A| = 0$ 。用高斯消去法求行列式的值所需运算量为 $O(n^3)$ 。如果应用舒尔余子式, 还可建立运算量为 $O(n^{2.8074})$ 的行列式快速求值算法。

矩阵求逆 如果 n 阶方阵 A 的行列式不为零, 那么 A 就称满秩矩阵 (或非奇异矩阵, 或可逆矩阵), 否则 A 就称为亏秩矩阵 (或奇异矩阵, 或不可逆矩阵)。一个满秩矩阵 A 有惟一的逆矩阵 A^{-1} , 使 $A^{-1}A = AA^{-1} = I$ 。通常矩阵求逆通过消元法来完成。在消元法中, 每步可以将对角线以外的元素消为零, 并使对角线元素为 1, 从而得到

$$N_n N_{n-1} \cdots N_1 A = I$$

其中 N_k 为在第 k 步将矩阵 $A^{(k)} = N_{k-1} \cdots N_2 N_1 A$ 的第 k 列变为 e_k (这里 e_k 是第 k 个分量为 1, 其余分量为 0 的列向量) 的矩阵。求逆的具体步骤如下进行, 设 $a_{ij}, a_{ik}, a_{kj}, a_{kk}$ 均指当时在 A 的相应元素位置上的数, 对于 $k = 1, 2, \dots, n$ 的每个 k , 依次做①计算 $d = 1/a_{kk}$, 并放在 a_{kk} 处; ②对于 $i \neq k$ 计算 $-da_{ik}$, 并放在 a_{ij} 处; ③当 i 和 j 均不等于 k 时计算 $a_{ij} + a_{ik}a_{kj}$, 并放在 a_{ij} 处; ④对 $j \neq k$ 计算 da_{kj} , 并放在 a_{kj} 处。最后在矩阵原来的位置上就得到 A 的逆矩阵 A^{-1} , 所需的运算量为 $O(n^3)$ 。

通过矩阵的分块以及快速矩阵乘法可以得到矩阵快速求逆算法。作法如下: 设 $n = 2^t$, 将 A 分作 4 个 $n/2$ 阶矩阵, 并令

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

按通常的分块求逆公式有 (假设 A_{11} 是满秩矩阵)

$$\begin{aligned} C_{22} &= (A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}, & C_{21} &= -C_{22}A_{21}A_{11}^{-1} \\ C_{12} &= -A_{11}^{-1}A_{12}C_{22}, & C_{11} &= A_{11}^{-1} - A_{11}^{-1}A_{12}C_{21} \end{aligned} \quad (5)$$

把式 (5) 中的计算式拆分为

$$\begin{aligned} B_1 &= A_{11}^{-1}, & B_2 &= A_{21}B_1, & B_3 &= B_1A_{12} \\ B_4 &= A_{21}B_3, & B_5 &= B_4 - A_{22}, & C_{22} &= -B_5^{-1} \\ C_{21} &= -C_{22}B_2, & C_{12} &= -B_3C_{22} \\ B_6 &= B_3C_{21}, & C_{11} &= B_1 - B_6 \end{aligned} \quad (6)$$

对于 A_{11}^{-1} 和 B_5^{-1} 的计算又可分为 $n/4$ 阶矩阵, 仍按式 (6) 重复进行, 如此递归地作下去, 最终化为 2 阶矩阵的求逆。如果矩阵乘法采用 S. Strassen 提出的算法, 则上述快速矩阵求逆算法的运算量为 $O(n^{2.8074})$ 。矩阵求逆除了上述消元法、分块法以外, 还有加边法和迭代法等。

三对角矩阵的求逆 在 n 阶矩阵 $A = [a_{ij}]$ 中, 如果 $j - i \geq 2$ 或 $i - j \geq 2$ 时, 有 $a_{ij} = 0$ (其中 $1 \leq i, j \leq n$), 则称 A 为三对角矩阵。三对角矩阵的求逆

可按下列步骤进行: 设 $l_0 = 1, c_n = a_{nn}$

$$l_i = \begin{vmatrix} a_{11} & a_{12} & & \\ a_{21} & a_{22} & \ddots & \\ & \ddots & \ddots & a_{i-1,i} \\ & & a_{i,i-1} & a_{ii} \end{vmatrix}$$

$$c_i = \begin{vmatrix} a_{ii} & a_{i,i+1} & & \\ a_{i+1,i} & a_{i+1,i+1} & \ddots & \\ & \ddots & \ddots & a_{n-1,n} \\ & & a_{n,n-1} & a_{nn} \end{vmatrix} \quad (1 \leq i \leq n)$$

$$u_{i-1} = a_{i,i-1} a_{i-1,i} \quad (2 \leq i \leq n)$$

递推地作

$$l_i = a_{ii} l_{i-1} - u_{i-1} l_{i-2} \quad i = 2, 3, \dots, n$$

计算

$$b_{ij} = \begin{cases} ((-1)^{i+j} c_{i+1} l_{j-1} \prod_{m=j+1}^i a_{m,m-1}) l_n^{-1}, & j \leq i \\ ((-1)^{i+j} c_{j+1} l_{i-1} \prod_{m=i}^{j-1} a_{m-1,m}) l_n^{-1}, & j > i \end{cases}$$

则有

$$A^{-1} = [b_{ij}]$$

上述三对角矩阵求逆算法的运算量为 $O(n^2)$ 。

参考文献

1. 冯康等. 数值计算方法. 北京: 国防工业出版社, 1978
2. 游兆永. 线性代数与多项式的快速算法. 上海: 上海科技出版社, 1980 (蒋增荣 成礼智)

juzhen tezheng zhi wenti shuzhi jiefu

矩阵特征值问题数值解法 (numerical solution of matrix eigenvalue problems)

指在数字计算机上, 研究如何采用有效的数值方法求矩阵特征值和特征向量的近似值的方法和过程。对元素为实数或复数的 $n \times n$ 维矩阵 A , 求数 λ 和对应的非零向量 x , 使 $Ax = \lambda x$, 这样的问题称为矩阵特征值问题, 也称代数特征值问题, λ 和 x 分别称为矩阵 A 的特征值和特征向量。矩阵特征值问题数值解常出现于动力系统和结构系统的振动问题, 以及物理学中临界值的确定。对于微分方程等连续系统的特征值问题, 若用离散化的数值方法求解也归结为矩阵特征值问题。此外, 在其他数值方法理论分析和讨论计算过程对舍入误差的稳定性问题时, 都与矩阵特征值问题有密切联系。

矩阵 A 的特征值 λ 是特征多项式 $P_n(\lambda) =$

$\det(\lambda I - A)$ 的根。其中 I 为 $n \times n$ 阶单位矩阵。传统方法是求 $P_n(\lambda) = 0$ 的根求出特征值 $\lambda_i (i = 1, \dots, n)$, 再求其相应特征向量。这种方法只能求低阶矩阵特征值, 对于 $n > 4$ 的高次多项式, 一般不能用有限次运算求出根的精确值, 直接用多项式求根, 工作量大且稳定性差。因此, 目前求矩阵特征值和特征向量的方法主要是向量迭代法和变换方法两类。

向量迭代法 不破坏原矩阵 A , 而是利用 A 对某些向量做运算产生迭代向量的求解方法, 多用来求矩阵的部分极端特征值和相应的特征向量。乘幂法和反幂法均属此类。

乘幂法 用来求矩阵按模最大特征值与对应特征向量的一种迭代法, 它以矩阵乘幂运算为主, 也称幂法, 设 n 阶矩阵 A 有一个完全的特征向量组, 其 n 个线性无关的特征向量为 $x^{(1)}, x^{(2)}, \dots, x^{(n)}$, 对应特征值按模大小满足条件: $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$ 。任取一个初始向量 $v_0 \neq 0$, 且 $v_0 = \sum_{i=1}^n \alpha_i x^{(i)}$ (设 $\alpha_1 \neq 0$), 于是

$$v_k = A^k v_0 = \lambda_1^k \left[\alpha_1 x^{(1)} + \sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1} \right)^k x^{(i)} \right]$$

由假设 $|\lambda_1| > |\lambda_2|$, 当 k 足够大时, $A^k v_0$ 除相差一个纯量因子外趋于 λ_1 所对应的特征向量, 实际计算时为避免出现溢出, 可采用规范化方法。最简单的幂法迭代格式如下:

取初始向量 $v_0 \neq 0$ ($\alpha_1 \neq 0$), 计算

$$u_k = A v_{k-1}, \quad m_k = \max(u_k)$$

$$v_k = \frac{u_k}{m_k} \quad (k = 1, 2, \dots)$$

式中 $\max(u_k)$ 是指 u_k 中绝对值最大的分量, 这时当 $k \rightarrow \infty$ 时, $v_k \rightarrow \frac{x^{(1)}}{\max(x^{(1)})}, \max(u_k) \rightarrow \lambda_1$ 。

反幂法 设 A 为非奇异矩阵, 为求 A 的按模最小特征值和相应特征向量而对 A^{-1} 使用乘幂法。 A^{-1} 的特征值次序是: $\left| \frac{1}{\lambda_n} \right| > \left| \frac{1}{\lambda_{n-1}} \right| \geq \dots \geq \left| \frac{1}{\lambda_1} \right|$, 取 v_0 为初始向量, 迭代格式为

$$A u_k = v_{k-1}, \quad m_k = \max(u_k)$$

$$v_k = \frac{u_k}{m_k}, \quad (k = 1, 2, \dots)$$

与幂法相似, $k \rightarrow \infty$ 时, $v_k \rightarrow \frac{x^{(n)}}{\max(x^{(n)})}, \max(u_k) \rightarrow$

$\frac{1}{\lambda_n}$, 反幂法的每次迭代要解一个 n 阶线性方程组。

这种原始的反幂法实际中很少使用,常用的反幂法是带原点位移的,位移取为已知近似特征值,用反幂法可求得相当精确的对应特征向量。

变换方法 利用一系列特殊的变换矩阵(初等下三角矩阵、平面旋转阵、豪斯霍尔德矩阵等),从矩阵 A 出发逐次进行相似变换,使变换后的矩阵序列趋于容易求得特征值的特殊形式的矩阵(如对角阵、三角阵、拟三角阵、三对角阵等)。这类方法多用于求中小规模矩阵的全部特征值,其优点是收敛速度快、计算结果可靠。雅可比方法、豪斯霍尔德方法、 LR 方法和 QR 方法均属此类,其中以 QR 方法最为有效,是目前求矩阵特征值最常用的算法。

雅可比方法 是用来计算实对称矩阵全部特征值及其对应特征向量的一种变换方法,其基本思想是通过一组平面旋转变换(正交相似变换)将对称矩阵 A 化为对角阵,从而求得全部特征值和特征向量。这种方法的优点是能在求特征值同时求得相当精确的近似正交规范特征向量,缺点是收敛速度较慢,目前较少使用。

豪斯霍尔德方法 是通过豪斯霍尔德变换将矩阵化为对称三对角矩阵,从而求得对称矩阵 A 的全部或部分特征值与特征向量。计算过程分 3 步。

第 1 步,利用豪斯霍尔德变换矩阵 P_r ,令 $A = A_0$, $A_r = P_r A_{r-1} P_r$ ($r=1, \dots, n-2$),经 $n-2$ 步将 A 变换为三对角矩阵 C ,其中 $P_r = I - \rho_r^{-1} W_r W_r^T$ ($r=1, \dots, n-2$), $W_r = (0, \dots, 0, a_{r+1,r}^{(r-1)} + \sigma_r, a_{r+2,r}^{(r-1)}, \dots, a_{n,r}^{(r-1)})^T$, $\sigma_r^2 = \sum_{i=r+2}^n (a_{i,r}^{(r-1)})^2$,取 σ_r 与 $a_{r+1,r}^{(r-1)}$ 同号,

$$\rho_r = \sigma_r (a_{r+1,r}^{(r-1)} + \sigma_r), C = \begin{bmatrix} c_1 & b_1 & & \\ b_1 & c_2 & & \\ & & \ddots & \\ & & & b_{n-1} & c_n \end{bmatrix}.$$

第 2 步,利用二分法计算 C 的特征值。

第 3 步,应用带近似特征值位移的反幂法求 C 的对应特征向量,再由第 1 步变换矩阵 P_1, \dots, P_{n-2} 求得 A 的对应特征向量。

豪斯霍尔德方法速度快,计算过程稳定,一般来说优于雅可比方法,适用于求解中小型矩阵的特征值问题。

QR 方法 是计算矩阵全部特征值最有效方法之一,它把矩阵 A 分解为 $A = QR$,其中 Q 为正交矩阵, R 为上三角矩阵。记 $A_1 = A$,进行迭代循环 $A_k = Q_k R_k, A_{k+1} = R_k Q_k$ ($k=1, 2, \dots$),矩阵序列 $\{A_k\}$ 中

每一个矩阵都与 A 相似,事实上, $A_k = Q_{k-1}^T A_{k-1} Q_{k-1} = Q_{k-1}^T \cdots Q_1^T A Q_1 \cdots Q_{k-1}$,在一定条件下, A_k 趋于块上三角矩阵(对角块为一阶和二阶子块,其中 2×2 阶子块给出 A 的一对复特征值),为减少计算量,总是先将 A 用正交相似变换化为上海森伯格矩阵 H ,再对 H 应用 QR 算法。为加快 QR 收敛过程,常采用带原点位移的 QR 算法,记第 k 次迭代的原点位移为 s_k ,将 $A_k - s_k I$ 进行 QR 分解,即 $A_k - s_k I = Q_k R_k$, $A_{k+1} = R_k Q_k + s_k I$ ($k=1, 2, \dots$),所得矩阵序列 $\{A_k\}$ 中每个矩阵也都与 A 相似,第 k 步位移 s_k 常取 A_k 的右下角元素 $a_{n,n}^{(k)}$ 或右下角二阶矩阵的特征值中最接近于 $a_{n,n}^{(k)}$ 者。带原点位移的 QR 算法的渐近收敛速度至少是 2 阶,当 A 对称时可达 3 阶。当 A 有复特征值时,带原点位移的 QR 算法不可能收敛,为此发展了对实矩阵的双重步 QR 算法。该法的基本思想是将带复原点位移的 QR 算法的相继两步合并成一个双重步,以避免重复运算。当求得近似特征值后,可用带位移的反幂法求对应的特征向量。 QR 方法具有收敛快、计算稳定的优点,已被广泛应用。

通常计算矩阵特征值问题可使用数学软件库 EISPACK。

广义特征值问题数值解法 A 和 B 是 $n \times n$ 阶矩阵,求数 λ 和向量 $x \neq 0$,使 $Ax = \lambda Bx$ 称为广义特征值问题,它是前面所述矩阵特征值问题的推广,当 B 非奇异,可化为 $(B^{-1}A)x = \lambda x$ 的矩阵特征值问题。当 A, B 对称且 B 正定时,可将 B 分解为 $B = LL^T$, L 为非奇异下三角阵,于是问题转化为求 $(L^{-1}AL^{-T})(L^Tx) = \lambda(L^Tx)$ 的矩阵特征值问题。目前已发展了一些直接求广义特征值问题的数值方法,常用的有广义雅可比法、广义豪斯霍尔德方法(也称行列式查找法)和 QZ 方法,它们分别是矩阵特征值问题雅可比方法、豪斯霍尔德方法和 QR 方法的推广,其中 QZ 方法是适合于求非对称矩阵广义特征值问题的最有效算法。

参考文献

曹志浩. 矩阵特征值问题. 上海: 上海科技出版社, 1980
(李庆扬 周树荃)

jufa moshi shibie fangfa

句法模式识别方法 (syntactic pattern recognition method) 一种基于结构关系,以形式语言为工具的模式描述和分析方法。

用计算机进行**模式识别**可以归纳为两种处理方法:统计(或称决策理论)方法和句法(或称结构)方法。统计方法是从模式中抽取一组能代表模式特性的测量值(称为特征),并用划分特征空间的办法来对模式进行分类。对于一些模式识别的问题,描述模式的结构信息具有重要意义。对模式的识别不仅要求能够把输入的模式指定到一个特定的类别,即把它分类,而且要对那些不至于把输入模式进行分类的特征加以描述。在这类问题中,所研究的模式通常是十分复杂的,需要的特征也很多。因此,用一些比较简单的子模式组成多级结构来描述一个复杂的模式,就成为人们研究的方向。为了表示多级结构信息,可以将一个模式用一些比较简单的子模式来描述,而每一个简单的子模式再用更加简单的子模式来描述,正如英语中的短语和句子由单词连接而成,单词又由字符连接而成一样。实质上,模式识别的句法方法从一个方面展示了模式的结构与语言的句法两者间的相似性质。为发挥句法方法的优越性选定的最简单的子模式,即“模式基元”,应比模式本身更容易识别。通过一组模式基元及其组合运算来描述模式结构的“语言”称为“模式描述语言”。由基元构成模式所遵循的那些规则,可以用生成模式描述语言的“文法”来确定。识别的过程是通过对描述给定模式的“句子”进行句法分析来确定该句子在句法(或文法)上是否正确。模式的描述,既类似于语言的构成,又并不局限于像语言那样,字符之间仅是左右的连接,而是可以扩展到高维的连接与描述,从而把对模式的研究转换为对模式基元构成的串、树或图的研究。

句法方法的构思始于20世纪70年代初。第一本系统地对句法方法加以论述的著作是美国Purdue大学傅京孙(K. S. Fu)教授于1974年在美国出版的《模式识别的句法方法》。该书奠定了句法模式识别方法的基础。以前句法方法并没有得到广泛的应用,主要问题在于用形式语言描述模式,即使对简单的模式,也必须采用“上下文有关文法”;而识别模式则要对这种类型的文法进行句法分析,而上下文有关文法的分析是十分复杂的,因而句法方法的应用受到限制。经过研究改进,着眼于把统计方法和句法方法两者有机地结合起来,以属性文法为基础,引入基元的属性,基元间、基元与子模式间的联接属性,规定一些“语义规则”,利用语言描述中语法和语义之间存在的一种“折衷关系”,通过增加语义描述的复杂程度而使语法描述变得简单,从而克

服了以往句法方法在应用中所遇到的困难,并且利用人工神经网络形成了一种“语义、句法方法”。由于吸收了统计方法与句法方法的优点,拓宽了句法方法的应用范围。这种方法已经成功地用于手写汉字识别这类结构信息起着重要作用的领域。

参考文献

1. Fu K S. Syntactic Methods in Pattern Recognition. Academic Press, Inc., 1974
2. Tai J W. Semantic Syntax-Directed Translation for Pictorial Pattern Recognition. Proc. ICPR. Munich, 1982
3. Tai J W. A Connectionist Syntactic Semantic Approach. Proc. Intl. Conf. on Information Sciences. Seoul, 1993

(戴汝为)

juxing jisuanji

巨型计算机 (supercomputer) 在一定时期可用的、运算速度最快、性能最高且技术最复杂的一种计算机。简称巨型机,又称超级计算机。它是计算机型谱中的最高档机型。巨型计算机主要用于解决大型计算机也难以解决的复杂问题。它是解决科技领域中某些巨大的挑战性问题的关键工具。

发展简史

现代巨型计算机起源于20世纪60年代末至70年代初。按照体系结构和技术水平的发展,巨型计算机已经历4代。

第一代巨型机是单指令流多数据流SIMD(参见**并行处理系统**)的**阵列处理机**AP,其典型代表是美国于1972年研制成功的ILLIAC-IV。它由64个处理机构成 8×8 阵列,这些处理机在一个控制部件的控制下同步并行工作。该机由分立元件组成,运算速度每秒近1亿次。

第二代巨型机是具有流水线结构的向量机VP(参见**计算机流水线**)。70年代中后期美国推出的Cray-1是这一代巨型计算机的标志。该机由高速中、小规模集成电路组成,有12条不同功能的流水线运算部件,分为4组,可并行流水工作,峰速达160 MFLOPS(按两条流水线计)。该机使巨型计算机首次成为商品,并走向市场。

第三代巨型机是多指令流多数据流MIMD的共享主存多处理机系统MP,以美国80年代中期到90年代初推出的Cray X-MP, Cray Y-MP系列为代表,具有2~16个处理机,紧密耦合共享主存,可多任务并发以解决用户的大型问题。每个处理机内又用了

多流水线向量技术,运算速度达每秒几十亿到近二百亿次。

第四代巨型机是大规模并行处理系统 MPP(参见大规模并行处理),它由成百上千,甚至上万个处理机互连而成,靠高度并行以获得超高性能。这一代巨型机起步于 80 年代初,利用超大规模集成电路,硬件实现相对容易一些,性能价格比高。较成功的机种有美国的 CM-5, nCUBE-X, Paragon, SP-2, T-3D 等。

分 类

巨型计算机有不同的分类方法。按用途,可分为通用巨型机和专用巨型机;按字长,可分为 32 位巨型机和 64 位巨型机;按所处理的数据是否是向量,可分为标量巨型机和向量巨型机;按所用的处理单元是否相同,可分为同构巨型机和异构巨型机;按体系结构,可分为上述四代巨型机,即 SIMD 阵列机、向量机、并行多处理机和大规模并行处理系统。图 1 是这 4 种机型的结构框图。

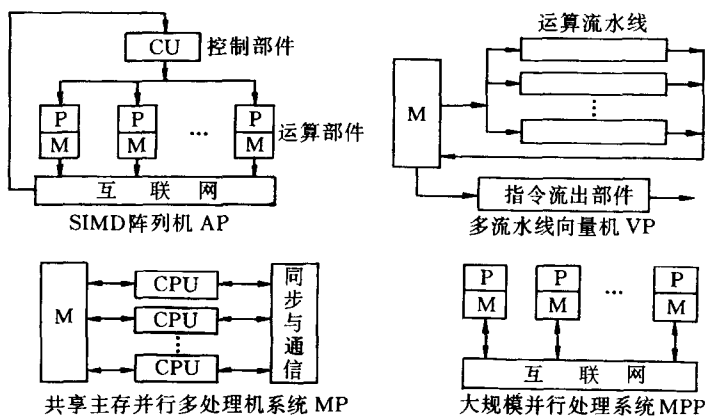


图 1 4 种巨型计算机

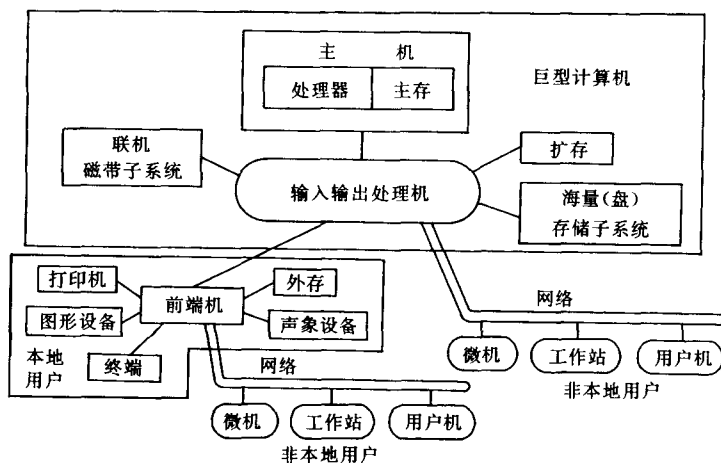


图 2 巨型计算机组成举例

组 成

巨型计算机通常由主机系统、输入输出系统和前端机系统组成,图 2 为巨型机组成的一个例子。

巨型计算机的核心是主机系统,它由高性能处理机和高速大容量主存构成。主机的任务是完成高

速运算。输入输出系统独立于主机,并行管理输入输出,向主机提供习题作业和运算所需的数据,接收主机的运算结果,并负责管理全部联机外围设备和接口通道。由于巨型机所加工数据的吞吐量很大,高速主存容量有限,一般还需要有半导体扩存和海

量存储子系统。对大规模数据处理的用户,常需联机磁带子系统或光盘子系统作为大量数据输入输出的媒体。前端机系统是本地用户使用巨型机的总管,用来准备程序和数据,向主机发送作业,接收主机的最终处理结果,对结果进行加工后输出。前端机一般采用小型计算机。非本地用户借助计算机网络使用巨型机,用户端的计算机(微型计算机、工作站等)可起前端机的作用,通过网络向巨型机发送作业和接收结果。

巨型计算机技术

(1) 体系结构 巨型机获得高处理速度的主要手段之一是并行技术。有两种并行,一是利用流水线原理,以时间重叠方式加工数据;二是多个处理部件以空间分布方式实现并行处理。在体系结构中要研究如何开发这两种并行,以构成一个合理、实用、高效的系统。多处理部件间的互连、同步和通信是技术难关。如何建立多处理部件互连拓扑与实际数学问题求解之间的映象,以充分发挥多处理部件的效率,是体系结构和并行算法相结合的重要研究课题。大容量存储器比运算器慢得多,如何使存储器和运算器的速度匹配是体系结构的一大难题。多级存储器是解决这一问题的传统途径。如何使输入输出能力和计算能力匹配是又一关键技术。

(2) 工程实现 巨型机获得高处理速度的另一个主要手段是提高计算机的时钟频率,即提高处理单元自身的处理速度。限制时钟频率提高的因素主要有两个,一是元器件的开关速度,二是元器件之间的连线长度。前者依赖于微电子技术的发展,后者依赖于器件集成度和高密度组装技术。巨型机的运算速度越高,其核心部分的物理尺寸应越小,这是高难度技术。巨型机庞大复杂,耗电量大,稳定均衡的供电是又一大难题。对于耗电量大而尺寸小的组装,散热是关键。

(3) 软件 巨型机的操作系统应是并行、分布式操作系统,具有批处理和一定的交互处理功能,它直接影响巨型机系统的稳定性和使用效率。大多数巨型机采用 UNIX 操作系统。并行编译器将高级语言编写的用户程序编译成能在巨型机的多个处理机或多条流水线上并行执行的程序,其关键是充分开发并行性,使程序得以高效地执行。对科学和工程计算,并行 FORTRAN 语言编译器是最主要的。软件工具和开发环境可以协助用户编制、优化、调试其并行程序,它们是帮助用户用好巨型机的重要软件。并行分布式数据库、各种科学计算的算法库、图形

库、高速网络软件、科学计算可视化软件等都是巨型机重要的软件。

(4) 并行算法 针对各种用户的复杂的实际问题,研究开发相应的并行算法是用好巨型计算机的又一个重要方面。算法和体系结构匹配可以提高巨型机的性能。

发展趋势

科学技术的发展和社会信息化对巨型计算机的计算能力不断提出新的要求,每秒万亿次运算的巨型计算机已经成为现实。微电子技术已在 1cm^2 的硅片上集成几百万个甚至更多的晶体管,商品化的 64 位微处理器的速度已达每秒几亿次浮点计算。以微处理器为基础构造巨型机已是大势所趋。这不但能使巨型机体积减小,价格下降,性能提高,而且容易做到规模可伸缩,小到几个处理机,大到上万个处理机,从而可构成一种平台的各档系列机。

参考文献

Kai Hwang. Advanced Computer Architecture: Parallelism, Scalability, Programmability. New York: McGraw-Hill Inc., 1993 (周兴铭)

juli tuxiang huoque yu fenxi

距离图像获取与分析 (range image acquisition and analysis)

用激光或者超声入射物体表面,通过反射延时直接测定物体表面距离所得到的图像以及用此图像求取物体形状的方法。

计算机视觉系统的输入数据一般是摄像机获取的灰度图像或彩色图像,图像上每一点的灰度(或彩色图像的相应测量值)反映了空间物体表面反射光的强度及颜色。三维计算机视觉系统可以通过两个(或更多个)摄像机同时获取多幅图像,从而计算出三维物体表面点离摄像机的距离,但计算复杂性高,鲁棒性也差。而通过距离图像获取装置得到的距离图像的每一点的值是物体表面上某点离观察仪器的距离,因此更容易用它来分析物体的形状。所以距离图像分析已成为计算机视觉的一个分支。

距离图像获取装置一般由四部分组成:光源(或超声源)、机械扫描部分、接收传感器以及计算机信息处理。光波或声波入射到物体表面,其反射波由接收传感器接收,由于光源一般是点光源或平面窄条光源,每次测量的只是物体表面一部分点的距离,所以用机械扫描装置控制光源的发射方向,以便顺序地测量物体表面上所有点的距离。

距离测量的原理一般分两种,一种称为飞行时间法,另一种称为三角测量原理。飞行时间法通过测量光波(或声波)从发射经物体表面反射再回到传感器的时间来计算距离。测量的原理也有两种,一种直接测量发射脉冲从发射到接收的时间差,称为脉冲法;另一种则对发射的波进行调制,通过测量发射波与接收波的相位差来计算时间差,称为相位法。由于计算机视觉系统观察的物体一般是周围的距离较近的物体,通常不超过几十米,光脉冲的传输时间只有几十纳秒,所以脉冲法对发送脉冲与接收器的要求很高,而用相位法实现较为容易。

三角测量原理也可用于距离图像的获取。如图1所示,为一束激光从A点出发,经物体表面C点反射,反射光经B点的光学透镜,聚焦在光敏器件组成的焦平面F上。在三角形ABC中,距离AB(称为基线长度)已知,角B可由聚焦光点在焦平面上的位置算出,角A为激光发射角,也是已知的,这样C点离光源的距离AC就可以通过三角形ABC算出。在实际应用时,A点发出的一般是一条平面窄缝光,C点是该平面与待测物体的交线,B处是一个摄像机,通过平面窄缝光的扫描(即改变A角)顺序地测量物体表面上各点的距离。

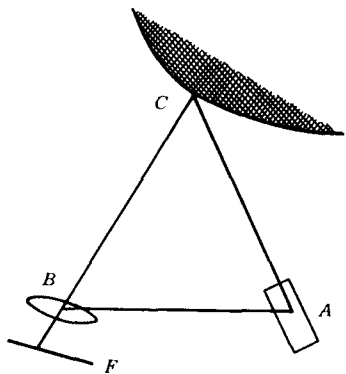


图1 距离图像的获取

距离图像的距离分辨率一般可达到毫米级,在有特殊用途时,还可以通过改善机械扫描装置的精度、光敏器件分辨率以及非线性补偿等方法进一步提高测距精度。

最早将距离图像用于计算机视觉的有 Stanford 国际研究所(SRI International)与日本大阪大学的 Shirai。前者研制了基于相位法的距离图像获取装置(1977年),测距范围1~5m,距离分辨率1cm;后者在1971年就研究了基于三角测量原理的距离图像获取。

距离图像给出的只是物体表面一大堆密集离散点的距离,经过数学变换后,也可以是这些点在三维空间的坐标。距离图像处理与分析研究从这些数据出发推导三维物体的表达与识别的方法。经过滤波消除噪音后,处理过程一般分为分割、曲面拟合和识别三个阶段。由于三维物体表面是由不同的连续曲面拼合成的(例如一个圆柱体是由两个平面与一个柱面组成),因此,需首先将距离图像上的点分割成几个区域,每个区域对应同一曲面。分割的方法一般是由距离图像出发,计算每一点所对应的物体表面的法向量、曲率等参数,由这些参数的变化确定不同曲面的接合点。曲面拟合则将属于同一曲面片的点拟合成一块曲面。拟合方法与曲面表达方式有关,一般的表达方式有代数曲面表示法或参数曲面表示法。经过拟合后,数据得到很大的压缩,物体由数个曲面片表示,每个曲面片只含几个参数。这些曲面数据以及曲面与曲面之间的相邻关系,构成了三维物体的表达。将该表达与数据库中已知物体的表达相比较,便可实现三维物体的识别。

距离图像处理目前仍然是三维计算机视觉领域一个活跃的研究课题。由于只有面对测距装置的物体表面上有数据,所以,数据是不完整的。另外,距离图像的噪声较大,这给三维物体距离图像的处理与分析带来很多困难。

由于距离图像的获取,省却了一般计算机视觉中由灰度图像计算三维物体距离的过程,节省了大量处理时间,因此,距离图像的获取与处理已较多地应用到工业部门,如在机器人装配、移动机器人导航、复杂曲面加工等方面都得到了应用。进一步提高距离图像的精度后,它还可应用于工业零件的无接触测量。

参考文献

Yoshiaki S. Three Dimensional Computer Vision. New York, Berlin, Heidelberg: Springer-Verlag, 1987

(马颂德)

juece zhichi xitong

决策支持系统 (decision support system, DSS) 以人机交互方式通过数据、模型、知识辅助决策者进行半结构化或非结构化决策的计算机应用系统。其主要目的是为决策者提供分析问题、建立模型、模拟决策过程和方案的环境,调用各种信息资源和分析工具,帮助决策者提高决策水平和质量。

决策按其性质可分为3类:①结构化决策 指对某一决策过程的环境及规则,能用确定的模型或

语言描述,以适当的算法产生决策方案,并能从多种方案中选择最优解的决策;②非结构化决策 指决策过程复杂,不可能用确定的模型和语言来描述其决策过程,更无所谓最优解的决策;③半结构化决策 介于以上二者之间的决策。这类决策可以建立适当的算法产生决策方案,但由于决策数据不完全或不精确,因而只能从相应的决策方案中得到较优的解。非结构化和半结构化决策一般用于一个组织的中、高层管理,其决策者一方面需要根据经验进行分析判断,另一方面也需要借助计算机为决策提供各种辅助信息,及时作出正确有效的决策。

决策一般分为4个步骤:①发现问题并形成决策目标,包括建立决策模型、拟定方案和确定效果度量。②用概率定量地描述每个方案所产生的各种结局的可能性。③决策人员对各种结局进行定量评价,一般用效用值来定量表示。效用值是有关决策人员根据个人才能、经验、风格以及所处环境条件等因素,对各种结局的价值所作的定量估计。④综合分析各方面信息,以最后决定方案的取舍,有时还要对方案作灵敏度分析,研究原始数据发生变化时对最优解的影响,决定对方案有较大影响的参量范围。决策往往不可能一次完成,而是一个迭代过程。决策可以借助于计算机决策支持系统来完成,即用计算机来辅助确定目标、拟定方案、分析评价以及模拟验证等工作。在此过程中,可用人机交互方式,由决策人员提供各种不同方案的参量并选择方案。

在决策过程中,为揭示决策问题的规律和本质,一般采用数学模型描述参与决策过程的诸变量之间

的约束关系。决策模型通常可用五元组 (M, S, P, A, V) 表示。其中, M 代表最优解,是决策人员希望达到的目标; S 为状态集,是系统所处的各种可能状态(s)的集合,即 $S = \{s_i | i = 1, 2, \dots, n\}$; P 是系统各种状态出现的概率,可表示为 $P(s)$; A 为方案集,说明决策的备选方案(a)所构成的集合,可用 $A = \{a_i | i = 1, 2, \dots, m\}$ 表示; V 为效用值,又称损益值,对不同方案估算出系统在不同状态下的结果或效益,是状态变量 s 和决策方案 a 的函数,用 $V(a, s)$ 表示。决策模型可有多种表示方法,如决策树法、表格法、矩阵法和各类规则(如关联、聚类、转移)等。

1970年,美国麻省理工学院的 S. S. Morton 等人根据计算机对决策的支持作用,提出决策支持系统的概念。1978年, S. S. Morton 和 P. G. Keen 在《决策支持系统:组织管理的前景》一书中,把决策支持系统(DSS)定义为“辅助管理者对半结构化问题的决策过程,支持而不是代替管理者的判断,提高决策的有效性而不是效率的计算机应用系统”。DSS 第一次国际学术讨论会于1981年举行。近年来,DSS 理论研究和实际系统的开发已有很大的进展,并出现了智能决策支持系统(IDSS)和群体决策支持系统(GDSS)。1985年, D. Owen 提出决策支持中心(DSC)的概念并已有系统实现。目前,已开发的决策支持系统主要用于企业预测和分析、计划和销售、研究与开发等职能部门,也有用于社会科学、宏观经济调控、市场和投资效益分析等方面的;此外,DSS 在军事、工程和区域规划等领域也有广泛的应用。

决策支持系统(DSS)基本结构如图1所示,主

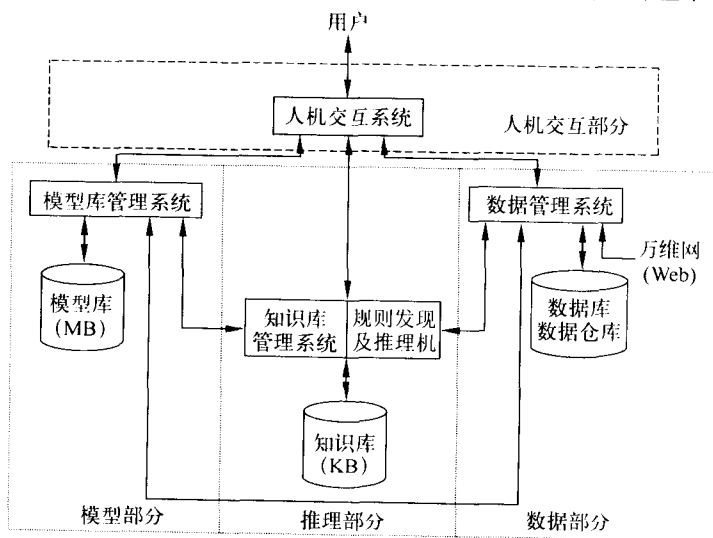


图1 决策支持系统基本结构

要由4个部分组成,即数据部分、模型部分、推理部分和人机交互部分。数据部分包括数据库(DB)、数据仓库和万维网(Web)数据源及其数据管理系统。模型部分包括模型库及其管理系统。推理部分由知识库、知识库管理系统和规则发现及推理机组成。人机交互部分是决策支持系统的人机交互界面,用以接收和检验用户请求,调用系统内部功能软件为决策服务,使模型运行、数据调用和知识推理达到有机地统一,有效地解决决策问题。

参考文献

1. Keen P G. Decision Support System: The Next Decade. Decision Support System, 1987,3: 253 ~ 265
2. Efreem G Mallach. 决策支持与数据仓库系统. 李昭智、李昭勇等译. 北京: 电子工业出版社, 2001

(孙志挥)

junshi zhihui xinxi xitong

军事指挥信息系统 (military command information system) 以计算机系统为基础,运用先进的信息技术进行获取、传输、处理和利用各种军事信息,并进行辅助决策,对部队实施指挥控制及战场管理的信息系统。通常所说的 C3I、C4I 和 C4ISR 等系统都是军事指挥系统的一些具体工程实现。

军事指挥信息系统的出现始于 20 世纪 50 年代。1953 年,美国开始研制以计算机为中心的自动化指挥系统,并于 1958 年率先建成了世界上第一个自动指挥控制系统,首次将地面警戒雷达、通信设备、电子计算机和显示器连接起来,实现了目标航迹绘制和其他数据显示的自动化。20 世纪 60 年代,随着远程武器的发展,特别是各种战略导弹和战略轰炸机的大量装备部队,指挥决策与作战行动执行单位之间可能彼此相隔数千千米甚至更远,单一的指挥控制系统已无法胜任现代战争的指挥与控制任务,无法实时地进行大量信息的传输。为了改变这种状况,美军又逐步建立了一批战略、战术指挥自动化系统,如全球军事指挥控制系统。从 20 世纪 70 年代初到 80 年代末,随着计算机技术的飞速发展,计算机渗透到指挥自动化系统乃至武器系统的各个节点、末梢以及一切与信息有关的环节,实现了信息采集、分析与方案制定、辅助决策等高层次信息处理活动的自动化。在这一时期内,美军研制了一系列军事指挥信息系统,如空中预警系统,空中指挥所系

统,地下指挥所系统,机动型战术空军的军事指挥信息系统,以及三军的联合战术信息分发系统。自 20 世纪 90 年代至今,军事指挥信息系统进入了综合一体化发展的新阶段。

军事指挥信息系统的组成从技术设备的角度考虑,可分为传感器系统、通信系统、数据处理系统、显示控制系统和技术保证系统。从系统功能的角度考虑,可分为指挥控制系统、预警探测系统、情报侦察系统、通信系统、电子战系统和战场支援保障系统。从战场信息流程的角度考虑,可分为信息获取与感知系统、信息传输与分发系统、信息分析与处理系统、信息开发与利用系统、信息安全与对抗和信息存储与显示系统。

信息获取与感知系统主要包括预警探测系统和情报侦察系统,解决的问题是要看得到、看得清,这是指挥决策的基础。预警探测系统是军事指挥信息系统最重要的实时信息源,它直接影响到观察、判断、决策、行动和整个军事行动的全过程。情报侦察系统主要功能是平时和战时搜集各种情报信息,为各级指挥员提供决策依据,为作战部队提供作战信息,这是取得战争胜利的重要保障。信息获取与感知系统将向全天候、全信息影像、多信源综合、微型化、智能化和强生存能力方向发展,形成无处不在的战场感知。

信息传输与分发系统是战场信息互通和共享的纽带,主要包括通信系统和数据链系统,解决的问题是连得通、传得快。通信系统由通信装备、设施和通信人员组成,用以组织通信联络、保障军队指挥的系统,其作用是把指挥系统诸要素联结成一个有机的整体。数据链系统是利用无线信道机,在各种飞机、水面舰艇、陆基武器及不同战术平台之间,实时、自动、保密地传输、分发和交换格式化技术数据的特殊数据通信系统,其作用是实现指挥系统与武器系统的无缝隙连接和战场信息共享。信息传输与分发系统将向一体化、宽带化、多功能、抗干扰方向发展。

信息分析与处理系统主要是解决算得精、判断准的问题,其核心是计算机及其软件,其关键是能够近乎实际地向各级指战员提供战场空间内敌我双方的态势信息,通过数据融合、情报融合与态势汇合等技术形成通用作战图像,使指挥员掌握战场态势,拟制作战方案。

信息开发与利用系统是军事指挥信息系统的核心,主要包括指挥控制系统,解决的是结合紧、用得好的问题。指挥控制系统是指军队的各级指挥所(中心)系统,指挥员通过数据处理、显示控制和辅助决策等对部队实施作战指挥和管理。

信息安全与对抗系统是所有其他各个系统的基础,主要作用是实现人机交互,保证指挥人员能对整个

作战进程全面地、不间断地、有效地指挥和控制。

未来的军事指挥信息系统将朝着数字化、网络化、自动化、智能化和一体化的方向发展,通过一体化建设,所有部队处在一个共同的指挥控制系统之下,拥有信息优势,使各军兵种能像一支单独的部队发挥作用,确保诸军兵种联合作战的协同指挥。

(李德毅)

K

katong donghua

卡通动画 (cartoon animation) 用计算机制作卡通风格动画的技术和过程。其分类有计算机辅助制作二维卡通动画与计算机三维卡通动画。

卡通原是英文 cartoon 的音译,指报纸或期刊上讲笑话或以幽默方式进行讽刺批评的漫画。在美术中则指一种粗糙的草图,其大小与最终完成的结果图一样。卡通动画是画出一系列表现动作的画面,由于人眼的视觉暂留作用,若以每秒 24 帧或更高的速度播放这些序列画面就可以形成连续运动的感觉。在卡通动画中角色的造型和动态一般比较简单并且夸张,其中以迪斯尼风格动画为代表。在我国动画片中还包括剪纸片、水墨动画以及折纸片,在视觉效果上具有独特的风格。

传统的卡通动画制作是相当费时费力的,一部长篇动画需要许多人员参与:导演、制片人、动画设计人员和动画辅助制作人员。为了提高效率和降低成本,动画制作要遵循一定的程序,具体包含 16 个步骤:剧本创作,故事板编写,声音记录,历程分解,造型设计,动画设计,中间画面绘制,Leica 卷片,试验线稿,整理,描线与着色,背景,检查,拍摄,样片和配音复制。计算机辅助制作二维卡通动画就是用计算机完成其中的某些步骤,如输入画面,着色,数字编辑,背景绘制,声音记录,试验线稿,曝光表和中间画面生成等。中间画面生成的原理是给出一个动画角色在第 M 幅和第 N 幅画面上的姿势,在它们中间的 $M+1$ 到 $N-1$ 画面由计算机计算生成出来,最常用的是线性或样条插值方法。

借助计算机图形技术还诞生一种三维卡通动画,在这里物体仍然采用光照模型进行真实绘制,但动画角色的造型和动作却比较简单和夸张,其中以《玩具总动员》为代表。近年来出现的**非真实感图形绘制技术**也为生成不同风格的二维和三维卡通动画提供了基础。

参考文献

1. 于金辉, 李一兵. 计算机动画原理与制作技术. 北京:清华大学出版社, 1995
2. Jinhui Yu & John W Patterson. Assessment

Criteria for 2D Shape Transformations in Animation. In: Proc. of Computer Animation'97, Geneva, IEEE Press, 1997, 103 ~ 112 (于金辉)

kaifa guocheng

开发过程 (development process) 软件开发人员所负责的一系列活动,其目的是依据合同成功地开发并交付软件。当要开发新的系统,或对已有的系统进行版本升级以及对已有系统进行有开发活动的移植时,都要涉及开发过程。

开发过程包含的活动有:需求分析、设计、编码、集成、测试、安装以及验收支持等,也可包含与系统有关的活动。开发过程同时还贯穿软件过程中其他过程的实施;开发人员通过**管理过程**管理开发过程;通过**剪裁过程**剪裁开发过程的活动;通过改进过程参与开发过程的管理(参见**软件过程**);通过基础设施过程在开发过程中建立基础设施(参见**软件过程**);通过**支持过程**的一些过程实施开发过程中的文档编制、联合评审、审计等。

以下是开发过程所涉及的活动,其中活动(1)是必须的,且是应当首先完成的,活动(2)至活动(13)是可选的,开发人员应当首先完成活动(1)的任务,然后根据活动(1)所产生的结果选择活动(2)至活动(13),完成开发过程。

(1) 本过程的实施准备 目的是为开发过程准备最基本的约定。其主要任务有:①依据合同及软件或系统的特点选择活动(2)至活动(13),所选择的的活动可重复或相关联,亦可循环交替;②制定本过程计划,计划中至少包含所需的内部标准、方法、工具、行为、责任和所使用的程序设计语言等;③指定各种文档的编制方式,安排其他各**支持过程**的实施方法。

(2) 系统需求分析 目的是根据合同分析系统的具体应用意图,完成系统需求。系统需求规约的内容有:①对系统的功能和性能的要求,包括安全、保密性;②人机界面、操作和维护需求;③设计方面的限制等。检查该系统需求规约是否达到以下要求:能否跟踪**获取过程**得到的系统需求并保持一致;

是否具备可测试性;是否具备系统结构设计条件;操作与维护是否可行等等。

(3) 系统结构设计 目的是建立一个高层的系统体系结构。该体系结构将系统需求按硬件、软件、人工操作这三个要素分为硬件配置项、软件配置项和人工操作项。检查该结构是否达到以下要求:能否与系统需求保持一致;所使用的方法是否符合标准;所确定的软件配置项需求是否可行;操作和维护是否可行等等。

(4) 软件需求分析 目的是确定软件需求及质量特性需求,并完成需求分析规约。软件需求分析规约的内容包括:①功能和性能需求;②外界与软件(即软件配置项)的接口;③合格需求;④安全需求,包括与操作、维护、环境等对人员的伤害有关的说明;⑤保密需求;⑥人机工程和人机界面需求;⑦数据定义和数据库需求;⑧现场安装及验收需求;⑨用户文档;⑩用户操作和运作需求;⑪用户维护需求等等。检查该软件需求是否达到以下要求:能否跟踪系统需求和系统结构;是否从外部与系统需求保持一致;软件需求内部是否具备一致性;测试覆盖程度是否达到要求;是否具备可测试性;操作和维护的可行性如何等等。

(5) 软件体系结构设计 目的是将软件需求分析规约转化为一个体系结构。其主要任务有:①定义并描述该结构的顶层部分;②为软件外部接口、数据库、软件各部件之间的接口作顶层设计;③设计用户手册的初级版本;④定义初步测试需求并制定软件集成计划等。检查该结构是否达到以下要求:能否与软件需求保持一致;结构内各部件是否具备一致性;所用设计方法与标准是否一致;是否具备进一步作详细设计的条件;操作与维护是否可行等等。

(6) 软件详细设计 目的是详细设计软件体系结构中的每个部件。主要任务有:①尽可能将每个部件细划为可进行编码、编译及测试的软件单元;②详细设计软件的外部接口、软件各部件之间的接口、各单元之间的接口;③详细设计数据库;④充实用户手册;⑤定义单元测试需求并制定单元测试计划;⑥充实集成测试需求并制定集成测试计划等等。检查该详细设计是否达到以下要求:能否与软件需求和体系结构保持一致;各部件以及各单元之间是否具备一致性;所使用的设计方法是否符合标准;是否可测试;是否具备易操作性和易维护性等。

(7) 软件编码和测试 主要任务有:①根据详细设计结果进行编码,提供测试数据,完成单元测试;②充实软件集成测试需求和集成计划;③充实用户手册等等。检查本活动是否达到以下要求:能否与软件需求和软件设计保持一致;在软件部件内部,各单元需求之间是否具备一致性;单元的测试覆盖程度是否达到要求;编码方法是否符合标准;是否具备软件集成及测试的条件;是否具备易操作性和易维护性等等。

(8) 软件集成 目的是制定计划,将上述活动得到的各软件单元和软件部件和从获取过程得到的软件集成为所需软件。主要任务有:①制定计划,计划中至少包括测试需求、步骤、数据、责任和时间进度表等内容;②按计划进行软件的集成;③充实用户手册;④针对合格需求制定合格测试集及步骤等等。检查本活动以及集成好的软件是否达到以下要求:能否与系统需求保持一致;内部是否具备一致性;测试所用的方法是否符合标准;是否符合预期结果;是否具备软件合格测试的条件;是否具备易操作性和易维护性等等。

(9) 软件合格测试 目的是完成软件的合格测试。依据合格需求进行合格测试,并充实用户手册。检查本活动是否达到以下要求:软件需求的测试覆盖度是否达到要求;是否符合预期结果;系统集成和测试是否可行;是否具备易操作性和易维护性等等。可能时,应支持审计工作,审计后应准备一套完整的软件,交付给活动(10)至活动(13)。

(10) 系统集成 目的是将交付来的软件集成到系统中。主要任务有:①将软件同有关硬件、人工操作项以及其他必要的系统集成成为一个系统;②为系统合格测试进行必要的准备,包括开发测试集和测试步骤。检查集成后的系统是否达到以下要求:系统需求的测试覆盖程度是否达到要求;所用测试方法是否符合标准;是否符合预期结果;是否具备系统合格测试的条件;是否具备易操作性和易维护性等等。

(11) 系统合格测试 主要任务是依据合格需求和合格测试计划进行系统合格测试。检查本活动是否达到以下要求:系统需求的测试覆盖是否达到要求;是否符合预期结果;是否具备易操作性和易维护性等等。可能时,应支持审计工作;并在审计结束后准备一套完整的软件以便进行软件安装或软件验收支持。

(12) 软件安装 目的是在目标环境中安装软

件。主要任务有:①制定安装计划,包括与软件安装有关的信息和资源;②实施软件安装,注意保证该软件和数据库能按合同的规定初始化、运行和终止。

(13) 软件验收支持 目的是支持获取者对软件的验收评审和测试。主要任务有:①支持验收评审和测试;②按合同交付文档及代码;③依据合同向获取者提供培训和支持。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes — IEEE Std 1074—1991

2. ISO /IEC 12207: 1995 Information Technology — Software — Part 1: Software Life-Cycle Process
(宿为民)

kaifang xitong

开放系统 (open system) 遵循公开定义的接口和协议,便于与其他系统互连的计算机系统。负责开发易移植操作系统接口 (POSIX) 标准的美国电气和电子工程师协会 IEEE P1003 工作组把开放系统定义为:按照开放的接口、服务和支持的规范而实现的系统。开放系统能使:①应用软件基本上不作修改就可实现在不同系统间的移植;②本地或远程系统的各应用间实现互操作;③各应用间可方便地实现交互作用。

开放系统所遵循的标准和规范必须是公开的,因为这是共同遵守、维护和适时更新的前提。IEEE P1003.0《POSIX 开放系统环境指南》把开放规范定义为:同国际标准一致的规范,并且以开放的、多数同意的过程进行维护,以便能随时适应新技术的发展。

开放系统环境的程序界面、人机界面、系统管理工具、互操作服务、通信服务和安全性等方面都是按所选用的公开标准(国际标准、工业标准或事实标准)实现的,从而使得在这种环境中的各计算机平台间能确保应用软件的可移植性、可剪裁性和可互操作性,在这种环境中的用户计算机资源能得到最大限度的保护。因此,标准是开放系统的依据;可移植性、可剪裁性和可互操作性是开放系统的特征;对用户利益的保护是开放系统的目的。

按照开放系统的目标所设计的计算机体系结构称为**开放体系结构**,用开放系统的概念和开放体系结构的计算机所构成的应用系统称为**开放应用系统 OAS**。

开放系统的形成与发展

回顾计算机科学技术和应用的发展历史,在网络计算尚未得到大规模的推广应用前,计算机系统主要由单机或由同类型计算机构成。有影响的计算机公司都拥有自己独有的操作系统,那时的产品虽然也宣称具有可互联、可共享等性能,但都以自我为中心。

随着计算机技术和通信技术的发展,以往那种独特的、排他的产品已无法适应信息互通、信息共享等需求。为了适应大规模推广计算机的应用和计算机网络化的需求,尤其是要把多台不同类型的、不同地点的计算机集成为一个系统,需要尽量提高软件的可复用性和可移植性,因此,提出了一系列的问题,即:如何能方便、有效地把多台不同类型的计算机连接在一起?如何确保集成后的系统能协调、高效地工作?如何能在今后的系统中,保证已有的硬、软件资源可继续使用和共享?

总之,大家认识到,今后计算机系统的发展策略必须从过去的以我为主的唯我型改变为遵循标准、开放兼容、资源共享的开放型。因此,可移植性、互操作性、易集成性等就成为开放系统所追求的基本目标了。为实现这一目标,首先是建立一个可以确保不同机器和系统间的互操作性、软件的可移植性和由低向高的资源可继承性(规模可伸缩性)的公开标准。有了这样一个公开标准后,用户就可能以最佳的性能价格比,从多个供应商那里优选计算机的硬、软件资源,构成与其他系统兼容和资源共享的系统。

为了能最终实现一个理想化的开放系统,用户从应用的角度出发,希望有一个标准化的计算机应用环境,在这个标准化了的计算应用环境中,用户可方便地实现不同应用系统间的互操作、应用系统的移植和剪裁以及计算资源的高度共享。从图 1 可见,一个理想的计算机应用环境,其中的每一部分都是遵照公开标准实现的。

世界上有许多组织机构能对标准的形成产生影响,如由政府组织的标准化机构、由特定应用领域的用户组织的标准化机构以及由工业界的生产厂家组织的标准化机构。ISO(国际标准化组织)、ANSI(美国国家标准学会)和我国的国家技术监督局等均为政府组织的标准化机构。在国际上,也存在一些由生产厂家组成的标准化机构,它们为了使自己的产品能与其他产品互连和兼容,根据自己的利益和能力制定某些标准。其中较有影响的有:UI, X-Open, OSF, COS 等。

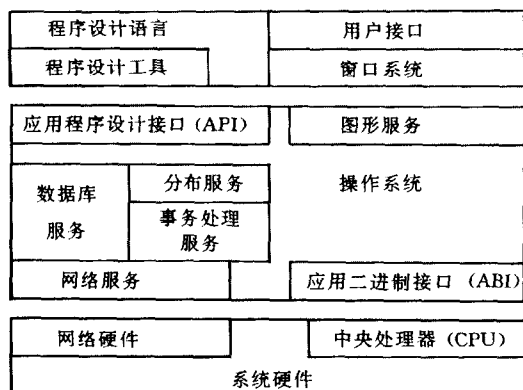


图 1 理想的计算机应用环境

开放系统的标准

国际上已公布或正在讨论和制定中的有关开放系统的标准大体上分为 3 类:

(1) 基础标准 负责制定功能标准的特别工作组 ISO/IEC JTC1/SGFS 在 TR 1000 技术报告中把基础标准定义为:为了在系统间交换信息而制定的一组单层或多层的规程和格式。

最有代表性和权威性的开放系统的基础标准是由国际标准化组织(ISO)所提出的**开放系统互连基准(参考)模型**。该模型定义了一种抽象结构,共由 7 层组成,即:物理层、数据链路层、网络层、运输层、会话层、表示层和应用层。该模型的特点是:①定义了一种能将异构系统互连的分层结构;②提供了控制互连系统间交互作用规则的标准框架;③定义了在不同计算机的同层之间实现通信的协议规程。

OSI 及相关标准是由 ISO 和国际电工委员会(IEC)两大国际标准组织的联合技术委员会 JTC1 负责制定的,由 JTC1 下属的有关分技术委员会(SC)和工作组(WG)具体负责开发这些标准,具体分工如下:

ISO/IEC JTC1/SC 6	系统间通信和信息交换
WG 1	数据链路层
WG 2	网络层
WG 3	物理层
WG 4	运输层
ISO/IEC JTC1/SC 18	文本和办公系统
WG 1	用户需求和 SC 18 管理
WG 3	文档体系结构
WG 4	文本交换系统
WG 5	内容体系结构

WG 8	文本描述和处理语言
WG 9	用户-系统接口和符号
ISO/IEC JTC1/SC 21	开放系统互连的信息检索、传送和管理
WG 1	OSI 体系结构
WG 3	数据库
WG 4	OSI 系统管理
WG 5	特定应用服务
WG 6	OSI 会话、表示和公共应用服务
WG 7	开放式分布处理(ODP)
ISO/IEC JTC1/SC 27	用于信息技术应用的公共安全技术
WG 1	密钥算法及应用
WG 2	公钥加密系统及使用模型
WG 3	通信体系结构中使用加密技术
ISO/IEC JTC1/SGFS	功能标准

OSI 及相关标准已超过 200 项,详细说明了协议规范和每一位的定义。

以下要介绍的 POSIX 属于基础标准。

(2) 功能标准 当用户在具体构建某个实际的开放系统时,应在基础标准的规范下,根据用户的实际需求和已具有的网络资源和能力,具体制定功能标准,或称为功能轮廓或规范文件。

OSI 在制定基础标准时,由于已预先考虑了各种不同的应用需求以及各式各样的网络结构,所以提供了多种功能选项。功能标准是基础标准的具体化,不能与基础标准相矛盾,只是根据具体情况对基础标准所允许的选项加以特定的选择。每个国家、企业或公司都可根据自己的具体情况对基础标准所提供的可选项进行设定,从而制定出各自的功能标准。

许多国家的政府部门均作出决策,要求其公共部门必须采购并使用 OSI 产品。美国政府机构的 OSI 功能轮廓 GOSIP 已作为联邦政府信息处理标准 FIPS 146,于 1988 年正式发布。

英国的中央计算机和电信局(CCTA)定义了名为 MUSIC 的开放系统应用结构框架,其中的 M,U,S,I,C 分别表示管理、用户接口、系统和应用接口、信息和数据服务、通信服务。

以下要介绍的 X-Open, OSF 等属于功能标准。

(3) 应用标准 当用户基于开放系统的概念,在具有开放体系结构的计算机系统上开发各自的开

放应用系统时,应按照应用标准,通过3种不同的抽象级别对该应用系统进行描述,即需求描述、过程描述和代码描述。

开放系统标准的制定涉及到计算机系统的各个组成部分,如中央处理器、操作系统、网络通信协议、数据库、各种服务功能和用户接口等。从用户需求出发,首先应解决的是用户接口和网络软、硬件的标准化问题,使用户可方便地、无阻碍地使用网络内的全部资源。从系统的技术实现出发,操作系统的标准化无疑是最核心的问题。

由于国际市场上的绝大多数计算机系统都使用UNIX操作系统,因此,UNIX也就成了实现开放系统的基本操作系统。在1984年,市场上曾存在着多种版本的UNIX。最有影响的是以下3种:由AT&T公司提供的UNIX System V,由Berkeley Software Distribution提供的BSD UNIX和由Microsoft公司提供的Xenix。如何统一这些已流行的UNIX系统成为实现开放系统的一个关键问题。

基于UNIX的开放系统规范

(1) POSIX

美国UNIX用户团体为统一UNIX标准曾编制了一个UNIX的统一规范,称为POSIX(可移植的操作系统接口)。1988年,经IEEE和ISO讨论后认定它为国际标准,并在原基础上进一步明确:POSIX是指由IEEE 1003.0~1003.9工作组开发的一套标准,目的是保证应用系统的可移植性。其中:

IEEE 1003.0工作组定义应用可移植性环境的元素、标准及其集成;

IEEE 1003.1工作组定义POSIX.1系统接口;

IEEE 1003.2工作组说明了POSIX.2 shell和工具;

IEEE 1003.3工作组定义POSIX.3测试和测试;

IEEE 1003.4工作组定义POSIX.4实时扩充;

IEEE 1003.8工作组制定POSIX.8网络文件系统

(NFS),提供透明的文件访问。

(2) OSF

开放软件基金会(OSF)是由135家计算机制造商和研究机构组成的非盈利组织,其目的是为开放软件环境制定一套应用环境规范(AES),并对按照这些规范开发的源程序发放许可证。

(3) UI和X-Open

UI是由AT&T公司和它的UNIX特许单位组成的专门对UNIX操作系统进行定义和控制的机构。1993年决定由X-Open继续原UI的标准化工作。X-Open成立于1984年,发起者以欧洲的计算机厂家为主,目的是促进UNIX操作系统标准化。它是一个非盈利性组织,任务是在ANSI,IEEE和ISO等标准化组织所正式公布的标准中选定自己的开放系统规范。

X-Open和OSF都不是制定基础标准的机构,它们只是从ANSI,IEEE和ISO等标准化组织正式公布的标准中,选择和决定开放系统需要的规范。表1扼要地给出了X-Open和OSF的开放系统

表1 X-Open和OSF的开放系统规范

	开放软件基金会 OSF (应用环境规范 AES)	X-Open (X-Open 可移植性指南 XPG 3)
操作系统 POSIX	ISO 9945-1 IEEE 1003.1 ISO 9945-2 IEEE 1003.2	ISO 9945-1 IEEE 1003.1 ISO 9945-2 IEEE 1003.2
用户接口	X-Window 系统 OSF/Motif	X-Window 系统
图形	GKS ISO 7943 GKS-3D ISO 8805 PHIGS ISO 9592	未作规定
网络	TCP/IP OSI XTI (X-Open 传输接口)	XTI (X-Open 传输接口)
数据管理	SQL ISO 9074	SQL ISO 9074 ISAM 终端接口源码交换
语言	FORTRAN ISO 1539 C ANSI X.3 159 PASCAL ISO 7185 COBOL ISO 1989 Ada ANSI/MIL 1815A LISP COMMON LISP BASIC	FORTRAN ISO 1539 C ANSI X.3 159 PASCAL ISO 7185 COBOL ISO 1989 Ada ANSI/MIL 1815A

规范。

(4) 在 POSIX 基础上, OSF 与 X-Open 的结合

为了真正建立一个基于 UNIX 操作系统的开放应用环境, UI (以及 X-Open) 和 OSF 都一直在促进建立符合 POSIX 规范的结合。具体的作法见图 2, 在共同遵循 POSIX 的基础上, OSF 的操作系统 (如 OSF/11.1) 和 UI 的操作系统 (如 SVR 4.2) 相互采用对方的基本技术。系统的操作系统是基于微核心软件 Mach 3.0 之上的; 在 Mach 之上建立大量面向各种主流机型 (如使用 OSF, BSD, UL-

TRIX, SVR4, VMS, MVS, DOS, OS/2, Windows 等的机型) 的服务软件; 由 OSF 提供中间件, 包括分布式计算环境 (OSF/MOTIF/DCE) 和分布式管理环境 (OSF/MOTIF/DME)。用户的应用软件是在中间件之上构造的。通过中间件确保应用软件的可移植性。X-Open 决定把它所开发的 XPG (X-Open 可移植性指南) 同分布式计算环境 DCE 相一致, 从而促进了 OSF 与 X-Open 的结合。但离用户所希望的、如图 1 所示那样的一个开放的应用环境还有很大的距离。

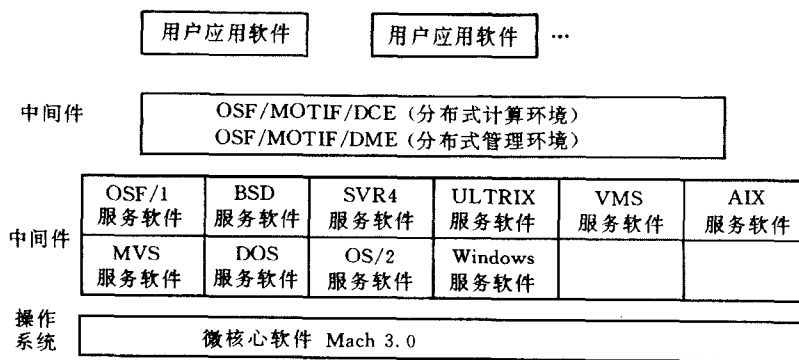


图 2 结合 OSF 与 X-Open 的具体做法

理想开放系统的实现是一个长期追求的目标, 是真正实现信息共享的必要前提。制定并遵循开放系统的规范和标准是实现开放系统的依据。由于实现开放系统的技术问题尚未全部解决, 各个集团和公司都在争夺制定各种标准的主导权和维护各自的利益, 各用户在淘汰 (或改造) 旧系统、转向开放系统时还存在习惯、资金和人力等因素, 因此开放系统的实现将是一个逐步完善的过程。

参考文献

1. Gary J Nutt. Open Systems. Prentice Hall, 1992

2. Pamela A Gray. Open Systems — A Business Strategy for the 1990s. McGraw-Hill, 1991 (汪成为)

kaifang xitong hulian jizhun (cankao) moxing (open system interconnection reference model) 由国际标准化组织 (ISO) 提出和定义的网络体系结构, 简称 OSI 基准 (参考) 模型 (ISO/OSI 7498), 如图 1 所示。

OSI 基准 (参考) 模型于 1977 年被 ISO 的 TC (技术委员会) 提出以来, ISO 又分别为各层制定了

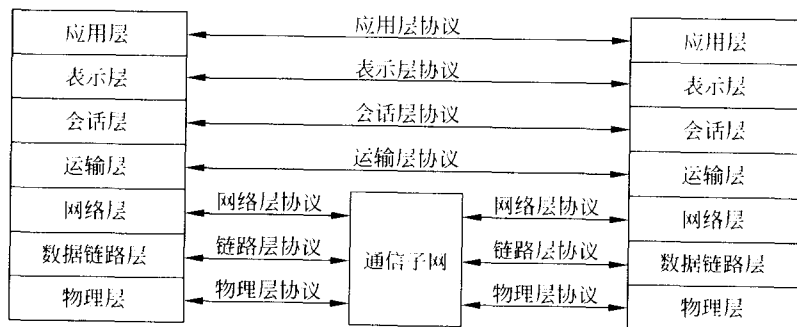


图 1 OSI 网络体系结构

协议标准,从而形成了完整的 OSI 网络体系结构。在 OSI 网络体系结构中,低层协议为相邻的高层协议提供相应的服务,高层协议作为低层协议的用户存在。OSI 网络体系结构的七层协议是:

(1) 物理层 物理层在通信信道上传输原始比特。物理层协议被设计来控制传输介质,从而提供传输介质以及和这些介质相连接的机械的、电气的接口。这些接口和传输介质必须保证发送和接收信号的同一性。

(2) 数据链路层 数据链路层协议加强物理层原始比特的传输功能。数据链路层把数据分装在不同的数据帧中发送,并处理接收方回送的确认帧。数据链路层通过在帧的前头和末尾附加上特殊的二进制编码来产生和识别帧界。数据链路层协议必须保证传输和接收的数据帧的正确性以及发送速度与接收速度的匹配。数据链路层协议还完成流量控制和出错处理工作。

(3) 网络层 网络层关系到对通信子网的运行控制。网络层负责选择从发送端传输一个数据包到达接收端的通信路径,起中继作用。网络层还负责通信子网中的分组,拥塞控制和记账等。路径选择方法通常有两种,即固定路径选择表方式和根据网络负载情况动态选择的方式。

网络层协议有面向连接的协议和无连接的协议两种。它们向高层提供面向连接的网络层服务和无连接的网络层协议(参见网络服务)。

(4) 运输层 运输层是 OSI 协议层次结构中最核心的一层。它把实际使用的物理网络与高层应用分开,提供发送端和接收端之间的高可靠性、低成本数据传输。运输协议为会话层提供面向连接的传输服务和无连接的传输服务。

为了提供性能可靠和价格合理的数据传输,运输层协议必须完成寻找接收端用户地址,提供面向连接服务时的建立连接、撤销连接以及流量控制和多路复用等工作。

(5) 会话层 会话层协议属于 OSI 基准(参考)模型的高层。它使用运输协议提供的可靠的端到端通信服务,并增加一些用户需要的附加功能和建立不同机器上的用户之

间的会话关系。

会话层协议为表示层提供同步服务,使得低层协议在发生了某种错误之后,会话层协议能返回到一个已知状态。会话层还为表示层提供活动管理功能。活动是一个由用户确定的具有逻辑意义的信息单位。会话层协议的另一个重要功能是数据交换。

(6) 表示层 表示层协议完成被传输数据的表示和解释工作,包括数据转换,数据加密,数据压缩等。表示层协议的主要功能有:①为用户提供执行会话层服务原语的手段;②提供描述复杂数据结构方法;③管理当前所需的数据结构集;④完成数据的内部格式与外部格式间的转换。

(7) 应用层 应用层包含大量人们普遍需要的协议。例如虚终端协议、文件传输系统、远程用户登录和电子数据交换以及电子邮件等。

在 OSI 网络体系结构中,高层协议用户通过服务访问点(SAP)和低层协议发生作用。服务访问点是服务使用者和服务提供者之间的界面。 N 层服务访问点记作 NSAP。同一层中的不同服务访问点用标识符来区别。服务访问点标识符被称为地址。相邻高层协议和低层协议的相互作用通过服务原语实现。服务原语是对服务提供者和服务使用者相互作用的最基本的行为的描述。它描述服务提供者和服务使用者一次最基本的交互作用的名称以及各参数的意义。

使用 OSI 网络体系结构时,除了物理层之外,网

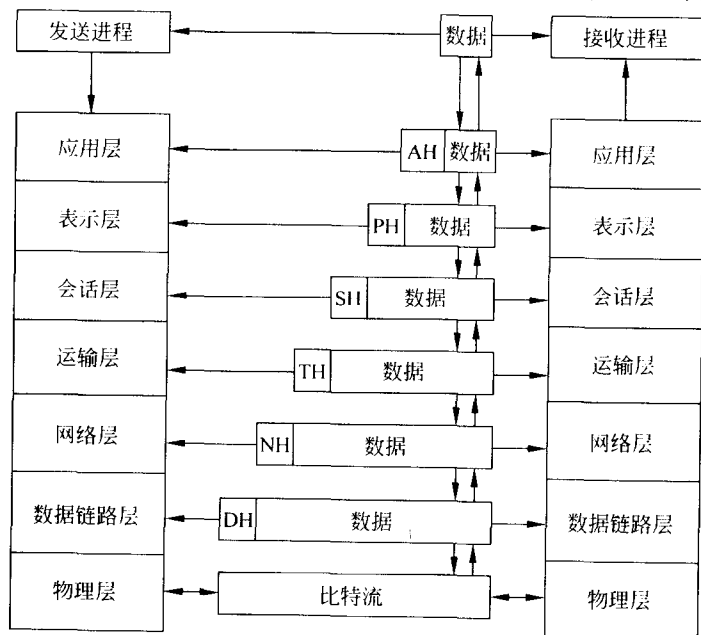


图2 OSI 网络体系结构中数据变化过程

络中数据的实际传输方向是垂直的。用户发送数据时,首先由发送进程把数据交给应用层,应用层在数据的前面加上该层有关控制和识别信息,再把它们交表示层。这一过程一直重复到物理层,并由传输介质把数据传送到接收端,在接收的进程所在的计算机中,信息向上传递时,各层的有关控制和识别信息被逐层剥去,最后,数据被送到接收进程。数据传输时数据变化过程如图2所示。

参考文献

1. ISO/OSI 7498. The OSI Reference Model. 1978
2. 张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999 (张尧学)

kaipanshi cidai qudongqi

开盘式磁带驱动器 (reel-to-reel tape drive)

计算机中以磁带为存储媒体,用于存储大量数据并能方便地更换的快启停磁带机设备。它有两个磁带盘,一为供带盘,又称文件盘;另一为固定盘,又称收带盘。由于两个带盘都能为使用者方便地装卸,所以称为开盘式磁带驱动器,通常也称开盘式磁带机。

从20世纪60年代初到80年代初,作为计算机外存储设备的开盘式磁带机得到了迅速发展,其机械结构、磁带驱动方式、缓冲机构与伺服系统、记录方式与读写电路、可靠性与互换性、自检功能与标准化等等方都有长足的进步,技术性能日臻完善和成熟,各种型号的开盘式磁带机名目繁多。

开盘式磁带机按其应用对象,有军用加固型和民用型两类,按磁带传动速度,有低速、中速、高速之分;按驱动方式有压轮式和单主动轮式;按磁带缓冲方式有摆杆式和真空积带箱式,而真空箱又有长、短积带箱之分;按记录方式有不归零制、调相制、成组编码;按机箱形式有台式和落地式;安装带方式又可分手工装带与自动穿带。开盘式磁带机的主要特点是快速启停。此外,存储容量大,可反复使用,易于复制和保存,维护使用简单,标准化程度高,信息便于交换,磁带价格便宜,因此直到现在仍然是计算机的重要存储设备之一。

目前开盘式磁带机采用的磁带宽度为12.5 mm (0.5 in),其技术性能与指标如下。

(1) 磁带速度 大致分为三档:低速为12.5 ~ 45 in/s (1 in = 25.4 mm);中速为75 ~ 125 in/s;高速为150 ~ 200 in/s。

(2) 启停距离 在各种速度下均为(4.8 ±

0.5) mm,因此带速越高,启停时间越短,例如带速为5 m/s (200 in/s) 时,启停时间仅1.3 ms。

(3) 记录密度 在逢1变化不归零制(NRZI)的记录方式下为800 b/in;在调相制(PE)方式下为1 600 b/in;在成组编码制(GCR)方式下为6 250 b/in。

(4) 记录间隔 在NRZI, PE的记录方式下为15.2 mm (0.6 in);在GCR方式下为7.6 mm (0.3 in)。

(5) 数据传输速率与每盘磁带的存储容量 以带速为75 in/s、记录密度为800b/in的中速磁带机为例,其数据传输速率为60 kB/s。若选用直径为10.5英寸的带盘,每盘磁带的总存储容量为20 MB。

(6) 磁头 为双缝组合型读写磁头,9道、带抹头。

(7) 带盘尺寸 有7英寸、8.5英寸、10.5英寸三种。其中7英寸、8.5英寸常用于低速磁带机,10.5英寸常用于中速、高速磁带机。10.5英寸的带盘容纳的磁带长为366 m (1 200 ft)。

磁带机通过磁带连接控制器(或格式器)与计算机连接,一般一个磁带控制器可连接4~8台磁带机。

开盘磁带机的品种型号繁多,性能指标差别也很大,但其基本组成却大体相近。

磁带机是一种机电一体化设备,一般包括机械和电路两大部分。机械部分主要有走带机构、带盘驱动机构、磁带缓冲部件、导向系统及磁头装置、电机与传感元器件。电路部分主要包括读写电路、伺服系统、控制逻辑与接口电路、电源等。原理可参见图1。

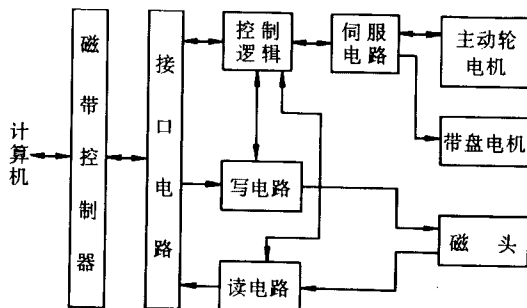


图1 磁带机原理图

对磁带机的基本要求有两点:一是带速稳定,张力不变;二是能准确无误地将数据写入磁带或从磁带上读出。为此需要伺服系统控制主动轮电机和两个带盘电机。与主动轮电机同轴的测速电机发出测

速信号,用以控制走带速度;为防止磁带打滑,中速、高速磁带机常采用真空主动轮。当磁带快速启停时,由于主动轮惯量小,带盘惯量大,带盘上的磁带线速度不能跟随主动轮线速度的瞬时变化。为此,在带盘与主动轮之间设置缓冲机构,以收容具有一定张力和适当长度的缓冲磁带。中速、高速磁带机的缓冲机构多采用真空积带箱形式,低速磁带机则多采用摆杆结构。

磁带机数据的写入、读出和抹除由读写组合磁头(带抹头)和读写电路等实现。

开盘式磁带机的品种型号很多,其结构形式有以下四种。

摆杆式低速磁带机 这种磁带机带速较低,所以用摆杆即可完成缓冲作用,摆杆在弹簧拉力的作用下使摆杆上的导柱以一定的张力张紧磁带。图2为摆杆式磁带机结构示意图,一种 Mod 7 的传动机构就属于这一类结构,其带速为 12.5 in/s,记录密度为 800 b/in(采用 NRZI 时)或 1 600 b/in(采用 PE 时),带盘为 7 英寸。

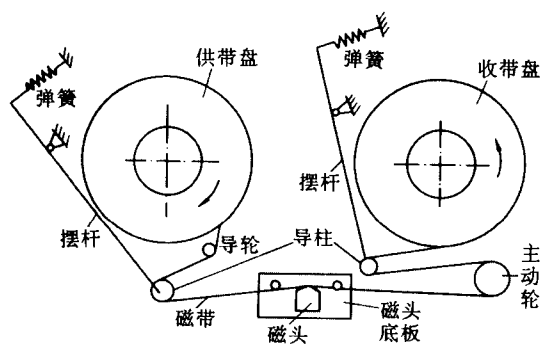


图2 摆杆式磁带机结构示意图

短真空积带箱中速磁带机 这种磁带机将缓冲磁带置于抽真空的积带箱中,形成带环,环下负压使磁带产生一定张力,以保证磁带的快速启停和走带的稳定。中速磁带机的带速一般为 75 in/s ~ 125 in/s,记录密度为 800 b/in(采用 NRZI 时)或 1 600 b/in(采用 PE 时),带盘尺寸为 10.5 英寸,可自动穿带、装带。如国产的 EDC-203, 204, 205, 206 型,美国的 TU 77, TU 78, 920, 9300 型等都属这种机型。

长真空积带箱中高速磁带机 磁带机由于带速很高,启停时的缓冲磁带增长,因此需要更深的积带箱,磁带机柜一般为落地式。典型结构如图3所示。它采用两个长条形抽真空的积带箱作为缓冲,因缓

冲带环增长影响快速启停性能,为减小带环质量,又增设了辅助积带箱。磁带机在机械结构和控制上采用空气轴承、真空主动轮、真空开关位置检测等技术,达到很高的技术水平。

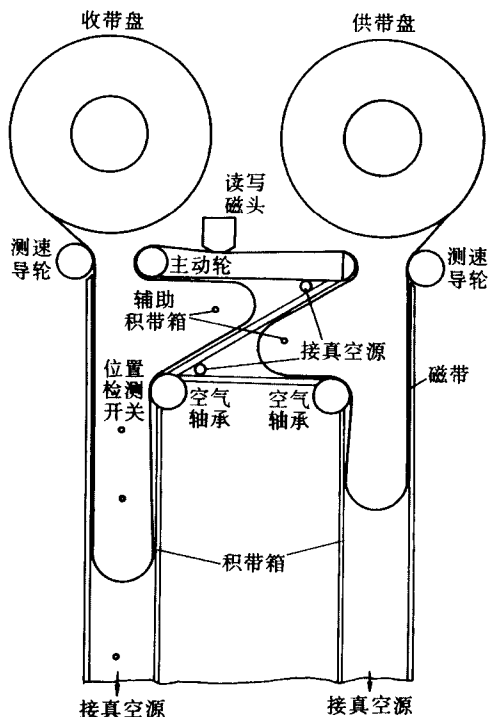


图3 高速磁带机结构示意图

IBM 3420 可作为这种机型的典型代表,其带速高达 200 in/s。由于带速高,导轮的惯性及导柱的摩擦力均能对磁带运动产生影响,为此采用了空气支承。此外还具有自动穿带(装带)、自动锁盘,并具有自动清洁器(倒带时自动保护磁头表面)、自动诊断功能等等。记录密度为 1 600 b/in(采用 PE 时)或 6 250 b/in(采用 GCR 时),数据传输速率较高。

半英寸流式磁带机 又称数据流磁带机,是 20 世纪 80 年代初才发展起来的,目前已得到广泛应用。它比传统的快速启停磁带机结构简单。可以有两种工作方式:一是流式(带速 100 in/s),二是启停式(带速 25 in/s)。它没有主动轮,用微型计算机控制。采用流式磁带写入数据时,设备自动插入 0.6 英寸的记录间隔,因而省去了为快启停而设置的复杂机构与电路。

半英寸流式磁带机通常具有标准磁带格式

(PE, 1 600 b/in)和接口电路,因此与同类型传统快启停磁带机有良好的互换性。这种机型由于结构简单,体积小,功能强,可靠性高,又有较强的自检能力,价格便宜,使用方便,颇受用户欢迎。

参考文献

郭平欣,姚锡珊,虞浦帆主编. 电子计算机外部设备原理. 北京:国防工业出版社,1986 (邓作夫)

kang elie huanjing jisuanji

抗恶劣环境计算机 (severe environment computer) 一种能在恶劣环境中正常运行的计算机。

恶劣环境包括物理、化学、生物及地理等因素。物理因素包括:高温、潮湿、海洋盐雾、沙尘、风雨、冰雪、高寒、冲击、振动、噪声、电磁、核辐射等;化学因素包括:带有腐蚀作用的各种有害化学气体,如 H_2S , CO , CO_2 , SO_2 等;生物因素包括:霉菌、蚁类、昆虫、鼠害等;地理因素是指高山、洞穴、森林、水下、空间等对计算机有影响的自然环境。为适应恶劣环境而对计算机进行加固的常用技术有:减振、散热、密封、屏蔽、涂层等。

抗恶劣环境计算机的历史大致可分为三个发展阶段。第一阶段(20世纪70年代及以前)——初始阶段。美国开始对武器系统的计算机进行加固以适应车载环境条件,如 SAGE 防空体系中的车载计算机。第二阶段(80年代)——发展阶段,抗恶劣环境计算机迅速发展,如美国 NORTON 公司的 750 R, DEC 公司的 PDP 11/34 M, ROLM 公司的 HAWK/32 M 等。第三阶段(90年代),抗恶劣环境计算机向高性能计算机方向发展,例如美国抗恶劣环境计算机厂商已采用 DEC 公司的 Alpha 芯片,研制高性能抗恶劣环境计算机。

抗恶劣环境计算机通常分为两类:第一类为**加固计算机**,即利用优选民用计算机系列产品,在工程技术上采取加固防护措施以适应恶劣环境的一类计算机。其环境可以是:工作温度 $0 \sim 50^\circ C$, 相对湿度 $10\% \sim 90\% RH$ (无凝露),海拔高度 $8\ 000\ ft = 0.3048\ m$,震动(正弦) $2\ g, 5 \sim 2\ 000\ Hz$,冲击加速度 $20\ g, 11\ ms$ 。加固计算机又可分为初级加固、半加固和加固3种。第二类为**全军用化计算机**,即专门为适应恶劣环境而设计、制造并符合军用标准条件和规范的一类计算机。其环境可为:工作温度 $-55 \sim +71^\circ C$,相对湿度 $95\% RH$ 以上,海拔高度 $70\ 000\ ft$,震动(正弦) $10\ g, 5 \sim 2\ 000\ Hz$,冲击加速度

$30\ g, 11\ ms$,电磁干扰满足 MIL-STD-810C 标准。

抗恶劣环境计算机主要应用领域有:工业过程控制系统,车、船、机、星载及野外作业系统,战略和战术武器系统,以及军事通信、指挥和控制系统等。

参考文献

国家标准总局. 国家标准《电工电子产品基本环境试验规程》. 北京:中国标准出版社,1983

(陆荣根)

kexue jisuan keshihua

科学计算可视化 (visualization in scientific computing) 将科学计算过程中及计算结果的数据转换为图形及图像显示在屏幕上的过程、方法与技术。

它综合运用计算机图形学、数字图像处理、计算机视觉、计算机辅助设计及人机交互技术等几个领域中的相关技术。既可以从复杂的多维数据中产生图形,又可以理解送入计算机中的图像数据。目前,这一技术的范围又有了扩展,它还包括工程计算数据的可视化及测量数据的可视化。

科学计算可视化的实现可以大大加快数据的处理过程,使目前每日每时都在生产的庞大数据得到有效的利用;可以在人与数据、人与人之间实现图像通信,而不是目前的文字通信或数字通信;可以使科学家们了解到在计算过程中发生了什么现象,并可改变参数,观察其影响,对计算过程实现引导和控制。总之,可使科学计算的环境和技术进一步现代化。

发展简史 计算机用于科学计算已有40多年的历史。在20世纪50年代和60年代,由于计算机的硬件、软件技术水平的限制,科学计算只能以批处理方式,大量输出数据的解释与理解所花费的时间往往是计算时间的十几倍甚至几十倍,这一阶段谈不上科学计算的可视化。70年代以后,虽然可以将科学计算的结果用二维图像表示,用绘图仪绘制出来,但仍然不能得到计算结果的直观、形象的整体概念,而且不能进行交互处理,只能被动地等待计算结果的输出。

近年来,随着科学技术的迅猛发展,待处理的数据量越来越大,来自巨型计算机、地球卫星、宇宙飞船、计算机断层摄影(CT)扫描仪及地震勘测的数据与日俱增,原有的数据处理及显示手段远远不能满足要求。在这样的形势下,1987年2月美国国家科学基金会在华盛顿召开了有关科学计算可视化的首次会议,与会者有从事计算机图形学、图像处理及各

种不同领域科学计算的专家,会议一致认为:将图形和图像技术应用于科学计算,这是一个全新的技术领域,会议将这一技术定名为科学计算可视化。

研究内容 科学计算可视化按其实现的功能,可以分为3个层次:①结果数据的后处理,即对计算数据或测量数据进行脱机处理,然后用图像显示出来。这一层次的功能对计算能力的需求相对较低。②中间数据或结果数据的实时跟踪处理及显示。③中间数据或结果数据的实时跟踪处理、显示及交互控制。这一层次的功能不仅能对数据进行实时跟踪显示,而且还可以交互式地修改原始数据、边界条件或其他参数,以使计算结果更为满意。

为了实现这3个层次的功能,科学计算可视化的主要研究内容有:①标量、向量、张量场的显示;②数值模拟和计算过程的交互控制和引导;③面向图形的程序设计环境;④高带宽网络及其协议;⑤用于图形和图像处理的向量和并行算法及特殊硬件结构。

应用领域 科学计算可视化的应用领域主要有:①计算流体力学;②有限元分析;③分子模型;④医学图像;⑤空间探测;⑥天体物理;⑦地球科学;⑧数学;⑨软件开发;⑩其他。

国内外发展情况概述 目前,国内仅有少数单位在进行科学计算结果数据可视化技术的研究,少数拥有较强计算能力的计算中心正开始应用这一技术实现有关领域的可视化,如气象、计算流体力学、天文、生物化学等。

在美国等发达国家的超级计算中心、国家实验室、著名大学,已在巨型计算机、光纤网、高档工作站的环境中实现了计算流体力学、有限元分析、医学图像、天体物理等领域计算结果的实时跟踪处理及显示,并正在研究科学计算过程的交互控制技术。目前,已有商品化的科学计算可视化软件,如AVS等。

随着互联网的广泛应用,共享网络中的数据资源成为迫切需要解决的问题。由于数据量十分巨大,数据类型繁多,逻辑关系复杂,国际上出现了用以表示海量数据不同类型及其逻辑关系的信息可视化技术。如果说科学计算可视化着重于空间数据的可视化,则信息可视化侧重于非空间数据的可视化。近期研究重点在于提出表示海量数据逻辑关系的可视化策略。

参考文献

1. McCormick B H, DeFanti T A, Brown M D.

Visualization in Scientific Computing. Computer Graphics, 1987, 21(6)

2. Nielson G M, Rosenblum L T. Visualization in Scientific Computing. IEEE Computer Society Press, 1990

3. 石教英,蔡文立. 科学计算可视化算法与系统. 北京:科学出版社,1996 (唐泽圣)

kebiancheng kongzhiqi

可编程控制器 (programmable controller, PC) 一种适合于工业环境下进行数字控制的专用电子装置。又称可编程逻辑控制器。它使用不易丢失记忆的存储器存放指令,主要执行诸如逻辑运算、顺序控制、计时、计数等功能。并可通过各种数字的或模拟的输入输出组件等,控制各种机械动作或工作顺序。

一台可编程控制器通常由带有存储单元的微处理器(主机)、输入输出模块、通信接口、编程设备以及电源等组成,如图1所示。其中微处理器用于执行各种指令,实现控制器的预定功能;存储器用于存放控制器的内部程序并为程序的执行提供运行空间;输入模块对输入信号进行检测,并转换成电平信号,供主机处理;输出模块将逻辑电平信号转换成高电平信号,如24 V DC, 24 V AC, 115 V AC, 220 V AC或4~20 mA DC信号,用来驱动各种马达、阀门、开关、指示灯以及有关设备;通信接口用于实现多台可编程控制器联网或与计算机通信,组成分散控制系统;编程器主要用来给可编程控制器输入和修改程序,也可用来监视可编程控制器的工作状态;电源组件给微处理器、存储器和输入输出模块供电。

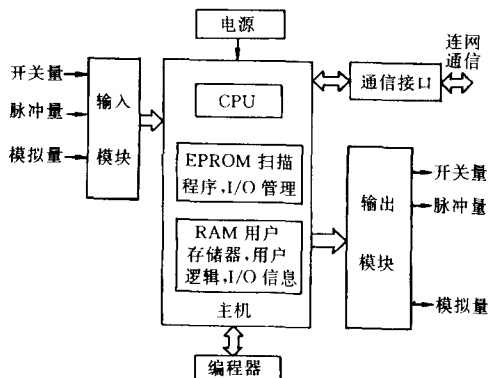


图1 可编程控制器构成图

可编程控制器主要用来实现程序控制,即按一定的时间顺序或根据某些给定条件(时间、状态)控制生产过程。早期,对于这种控制问题,通常采用继电器逻辑电路。相比之下,可编程控制器具有下述优点:

(1) 程序控制系统组成极为简单 根据控制要求编制程序并存入存储器,接上相应的输入输出信号线,就组成了一个程序控制系统。若要改变控制系统,只需修改程序,无需重新设计线路或修改导线连接等,具有良好的灵活性。

(2) 可编程控制器的程序编制采用直观、易懂的梯形图、流程图或指令表,编程简单,修改控制系统方便。

(3) 可靠性高 因为可编程控制器采用了集成电路,没有继电器那样的机械触点和内部接线。同时,在可编程控制器的设计中采用了冗余技术,如冗余处理器、冗余输入输出等,因此,它的平均故障间隔时间(MTBF)已大于数万小时,而平均修复时间(MTTR)则少于10分钟。另外,可编程控制器对使用环境要求不高,无需恒温装置。

(4) 控制功能强 可编程控制器的输入输出点数可达1000点以上,具有复杂的数据和信息处理能力,同时,增加了模拟量控制和计算机通信功能。这样,就可以实现整个生产过程自动化,适应现代工业中工厂自动化(FA)和计算机集成制造系统(CIMS)的需要。

在机械制造业和其他工业生产过程中,大多有程序控制和各种安全保护连锁控制系统。例如工业生产开车停车过程、间歇生产过程、数控机床、汽车生产装配过程等,都要求各种程序控制。因此,可

编程控制器在现代国民经济生产中有广泛的应用。

参考文献

朱善君等. 可编程序控制系统原理·应用·维护. 北京: 清华大学出版社, 1992 (王树青)

kebiancheng zhidu cunchuqi xinbian

可编程只读存储器芯片 (programmable read only memory chip)

一种可由用户自己用电的“硬”编程方法一次性地写入数据的半导体只读存储器芯片。简称 PROM 芯片。PROM 芯片产品出厂时是未经编程的,所存储内容是全0或全1。用户可根据自己的使用要求用电的方法编程,将串联于相应存储单元的连接熔丝熔断开路或将相应的连接用PN结二极管熔合短路以区别存储内容为1或0。PROM 芯片常称为熔丝 PROM 芯片,该芯片编程是一次性的,一经编程就不能修改存储内容。

早期的 PROM 芯片主要是用快速双极型工艺制作的,1980年以后 PROM 应用产品中也包括了CMOS型。双极 PROM 芯片的连接熔丝有两种类型:一种是使用熔断丝,大多采用铝、镍铬、钛钨、铂硅化物或多晶硅等材料;另一种是使用PN结二极管,采用肖特基TTL晶片生产工艺,利用扩散方法制造,PN结熔合短路是由于硅体内使用了一种稳定的硅铝易熔材料。图1为2×2位PROM存储单元阵列,图1(a)为熔丝熔断开路型,图1(b)为PN结二极管熔合短路型。图中存储单元已被编程写入数据,图1(a)中右上角熔丝处于开路状态,图1(b)中除右上角外,其他3个二极管处于短路状态。

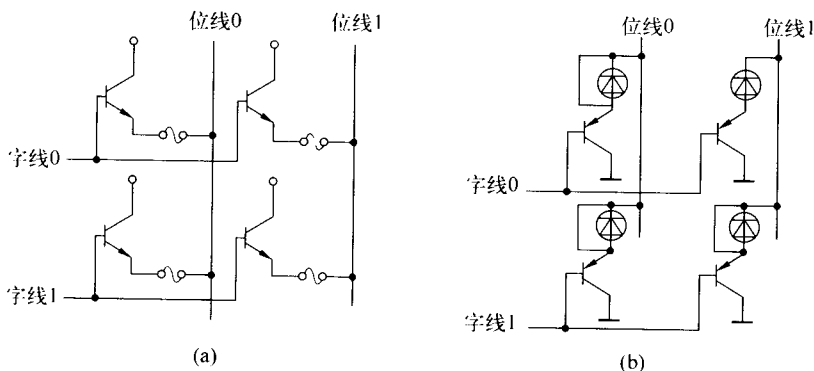


图1 2×2位PROM存储单元阵列

图2为商品16 kb 寄存器型 PROM 电路框图,其熔丝材料为熔断型多晶硅。该电路由地址寄存器、译码器、 128×128 存储阵列和相应输出缓冲器组成,特点是增加了输入锁存地址寄存器。PROM 主要用于微程序控制存储、微处理器程序存储,以及查表、字符发生和代码变换等。

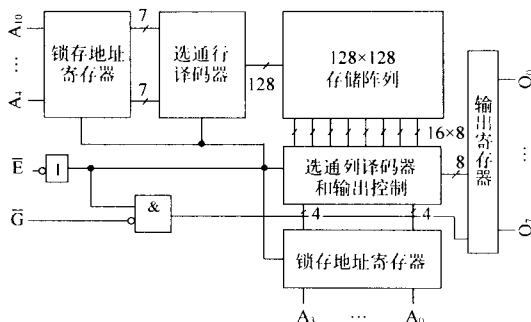


图2 16kb CMOS PROM 芯片电路框图

随着电子产品的发展,1999 年已实现批量生产的 PPRM 芯片(产品-可编程只读存储器芯片)产品获得了市场的广泛认可。与标准的掩模 ROM 芯片相比,PPROM 芯片的交货时间降低到掩模 ROM 芯片的 14%~33%,但其价格和掩模 ROM 芯片相当。写入 PPRM 的程序数据不需要使用掩模板,可以在集成电路封装以后进行,允许在生产的最末时刻改变存储器中的程序数据,因而对掩模 ROM 技术提出了挑战。具有类似于 PPRM 芯片性能的新型 PROM 芯片产品还有 2003 年实现了批量生产并获得应用的 Matrix 3-D 存储器芯片和灵活 ROM 芯片,它们也在抢占掩模 ROM 芯片的市场份额。

参考文献

Metzger I. R. A 16K CMOS PROM with Polysilicon Fusible Links. IEEE Journal of Solid State Circuits, 1983, SC-18(5):562 (时万春)

keca biancheng zhidu cunchuqi xinpian
可擦编程只读存储器芯片(erasable programmable read only memory chip) 一种存储内容可由用户用电的方法写入,并能用紫外线照射的方法擦除后再次写入的半导体只读存储器芯片。简称 EPROM 芯片。EPROM 芯片通常用 12V (或 20V) 的编程引脚在电路系统中写入数据,擦除时需将其从系统中取出并置于紫外线环境下,让光

线通过陶瓷封装的石英窗口照射芯片 20~30 min,可同时擦除全部原有信息。这种存储器芯片常称为紫外线 EPROM 芯片,记为 UV EPROM 芯片。习惯上还将功能上与 PROM 芯片等同,允许用户使用电的“软”编程的方法写入数据,但不能擦除的只读存储器芯片也归为 EPROM 芯片,称一次可擦可编程只读存储器芯片(EPROM 芯片),记为 OTP EPROM 芯片。OTP EPROM 芯片的最原始型式就是将 UV EPROM 芯片装在无石英窗口的塑料封装中。

UV EPROM 芯片是 1971 年开始出现的,早期产品主要使用 NMOS 工艺制造,其存储阵列与只读存储器(ROM)一样,且每个存储单元由一个晶体管组成。1980 年生产出 16 kb CMOS 型 UV EPROM 芯片,1986 年已发展到 1 Mb 并逐渐取代 NMOS EPROM 芯片。1990 年集成规模已达 16 Mb,CMOS EPROM 芯片产品已达 EPROM 芯片总量的 84%,成为 EPROM 芯片的主导工艺。

图1为典型 1 Mb CMOS EPROM 芯片电路框图。该电路可选择两种字位结构:字宽结构 128 kb × 8 和字宽结构 64 kb × 16。该电路型式的典型写入数据(编程)电压 V_{pp} 为 12.5V,读数电压 $V_{cc} = 5V$,存取时间范围为 100~300 ns。该电路特别增加的输出使能 OE 端可用于消除多重总线系统中的总线竞争。该电路有三个控制引脚,即芯片使能 \overline{CE} 、编程 PGM 和输出使能 \overline{OE} 。它们的组合可使芯片选择下述状态:读出、输出禁止、芯片待用、编程、编程验证和编程禁止。当 A_9 选择 V_{pp} 时,芯片处于电帖码方式,可以方便地读出生产厂家和器件类型代码。

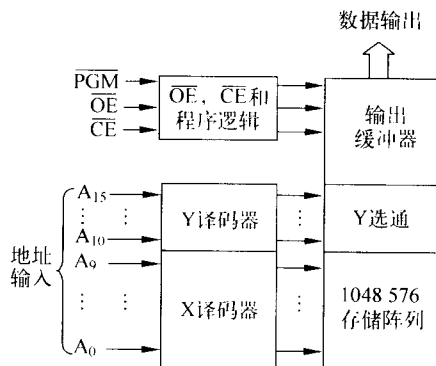


图1 典型 1 Mb CMOS EPROM 芯片电路框图

EPROM 芯片广泛用于办公自动化设备、医疗和通信设备,包括其中的快速语言翻译程序、光学字符识别系统和在线帧存储应用。工业上用于只读存储

器的样件开发和工程应用的替换。UV EPROM 芯片的缺点是成本较高,随存储容量增大其读数性能变坏。OTP EPROM 芯片为塑料封装,成本低,特别适用于快速周转应用,但由于它只能进行一次写入数据(编程),且缺乏测试空白芯片的技术和手段,推广应用有一定局限。

参考文献

Luecke G, Mize J P, Carr W N. Semiconductor Memory Design and Application. McGraw-Hill Book Company, 1973
(时万春)

kechuandai jisuan

可穿戴计算(wearable computing) 将计算机穿戴在用户身上进行信息处理和通信的技术。用这种技术所研制的计算机称为**可穿戴计算机**。用户可在移动的环境中使用这种计算机,并且可以持续地和它进行交互。

可穿戴计算的概念早在 20 世纪 50 年代就被提出来了,但相关技术的研究与应用是在 90 年代以后才随着电子、计算机和通信技术的发展而迅速发展起来的。目前可穿戴计算机的应用正逐步从军事向更广阔的领域拓展,如设备安装、采掘工业、野外勘探、医疗监测、新闻采访,直至走进人们的日常生活。

可穿戴计算不仅是将计算机微型化并与服饰融为一体,而且实现了人机的紧密结合,体现了以人为本的理念。可穿戴计算机一般配有头戴式或眼镜式超微型显示器,主机和其他外设嵌在衣物中,采用多模式接口和多通道传感技术,特别适合在户外和移动中使用。虽然不同用途的可穿戴计算机在构成、功能、形态等方面很不相同,但有其共同的特征,包括:①**人机一体** 作为一种最自然的携带方式,可穿戴是其本质,需要在人机工程学的指导下,通过**片上系统(SoC)体系结构、嵌入式操作系统**等技术使计算机主机微型化,并使其具有多端口和高性能 I/O;②**操作方便** 人机自然交互是其可用的关键,需要通过多模式接口和多通道传感技术最终将人的双手从操作键盘、鼠标中解放出来,目前单手键盘是一种最可靠而且实用的操作模式;③**移动中用** 用户可在移动状态下使用是其突出特点,要求能够适应人的活动并且作为一个移动结点随时上网,需要建立以人为结点的移动计算系统;④**持续工作** 可持续性是其区别于其他移动设备的重要特征,总是处于工作状态的计算机还可通过上下文感知等技术主动感知用户的环境并自动作出响应。

可穿戴计算目前已经形成一个新的热点研究领域,需要解决的一些特殊的关键技术有:①**片上系统体系结构** 把计算机主机的硬件集成到一个低功耗、高性能的芯片里;②**多端口、高性能 I/O** 用微小型的主机连接许多外设;③**嵌入式操作系统** 实时的和微内核的操作系统,并具有强大的处理多外设的能力;④**外设的设计** 所有外设需要是高性能、小体积、低功耗,符合人体特征,有利健康,安全可靠;⑤**无线连接技术** 分布于身体各部位的多个部件之间以近距离无线通信取代连线;⑥**无线自组网络** 没有固定路由器支持的各结点以任意方式移动并动态连接;⑦**人机交互** 解决用户与计算机之间的交互问题以及支持用户对环境的感知能力,包括听觉和视觉方面的识别技术、传感技术、信息融合技术和上下文感知技术;⑧**移动数据库技术** 支持用户在移动时的网络断接频繁、条件多样、信道不对称,计算部件伸缩范围大等条件下的数据库管理技术。

可穿戴计算机在不断更新换代,可以预见的是:

①由于专用芯片和专用设备的改进,可穿戴计算机将越来越小,交互技术的提高使得使用也越来越方便;②穿戴方式将有很大的改进,可以放在手表里、眼镜上甚至植入皮下;③将出现许多可选的专用设备和软件,使系统的构成能做到随心所欲;④可穿戴计算机的外设之间将实现无线互连。(史元春)

kejisuan hanshu

可计算函数(computable function) 能够在抽象计算机上计算其值的函数。1936 年,几位数理逻辑学家从不同角度给出了可计算函数的精确描述,并且证明,这些可计算函数类相同。J. C. Shepherdson, H. E. Sturgis 于 1963 年给出了无限寄存器机器(URM)。用 URM 定义的可计算函数类与用其他方法定义的可计算函数类相同。URM 模型比图灵机模型易于理解。下面介绍 URM 可计算模型。

URM 有无穷多寄存器 R_1, R_2, R_3, \dots 。任何时刻,每个寄存器均含一个自然数,用 r_n 表示 R_n 中的自然数。寄存器的内容可以通过指令改变。有穷多条指令构成一程序。URM 指令有 4 种(下面的 N 为自然数集合):

(1) 清零指令 $Z(n), n \in N, Z(n)$ 将 R_n 的内容变为 0,其余不变。

(2) 后继指令 $S(n), n \in N, Z(n)$ 将 R_n 的内容加 1,其余不变。

(3) 传输指令 $T(m, n), m, n \in N, T(m, n)$ 将 R_m 的内容 r_m 移入 R_n 中, 其余不变 (包括 R_m 也不变)。

(4) 跳转指令 $J(m, n, q), m, n, q \in N$, 在程序 P 中遇到指令 $J(m, n, q)$ 时就比较 R_m, R_n 的内容, 如果 $r_m = r_n$, 则 URM 进行 P 的第 q 条指令。如果 $r_m \neq r_n$, 则 URM 依次执行下一条指令。如果因 P 少于 q 条指令而不可能跳转, 则 URM 停止运算。

要 URM 执行一计算, 必须给出程序 P 并设置初态。所谓初态, 指寄存器中初始自然数序列。

假设 P 是一个 URM 程序, $a_1, \dots, a_n, b \in N$, 记号 $P(a_1, \dots, a_n, 0, 0, \dots)$ (简记为 $P(a_1, \dots, a_n)$) 表示将程序 P 作用于寄存器初态为 $a_1, \dots, a_n, 0, 0, \dots$ 的自然数序列。如果计算 $P(a_1, \dots, a_n)$ 终将停止, 停止时, R_1 中内容为 b , 则称 $P(a_1, \dots, a_n)$ 收敛于 b , 记为 $P(a_1, \dots, a_n) \downarrow b$ 。设 f 是一个从 N^n 到 N 的部分函数, 如果对任意 $a_1, \dots, a_n, b \in N$, $P(a_1, \dots, a_n) \downarrow b$ 当且仅当 $(a_1, \dots, a_n) \in \text{dom}(f)$ 且 $f(a_1, \dots, a_n) = b$, 则称程序 P 计算了 f 。

如果存在一个 URM 程序计算 f , 则称函数 f 是 URM 可计算的。

例如, 函数 $x + y$ 可用 URM 程序

```

11 J(3, 2, 5)
12 S(1)
13 S(3)
14 J(1, 1, 1)

```

来计算。

寄存器的初态为: $x, y, 0, 0, \dots$; 程序不断地将 1 加入 R_1 中, 用 R_3 来记录加 1 的次数, 一旦加 1 次数为 y , 则停机, 此时, R_1 中的数恰好为 $x + y$ 。因此, $x + y$ 是 URM 可计算的。

函数

$$f(x) = \begin{cases} x/2, & \text{如果 } x \text{ 是偶数} \\ \text{无定义}, & \text{如果 } x \text{ 是奇数} \end{cases}$$

是 URM 可计算的。令初态为 $x, 0, 0, \dots$ 执行程序如下:

```

11 J(1, 2, 6)
12 S(3)
13 S(2)
14 S(2)
15 J(1, 1, 1)
16 T(3, 1)

```

其中, 当 16 执行完后, 因无后续指令, 自行终止。

20 世纪 30 年代, 可计算函数概念的出现, 标志着一个新的数学分支的诞生, 它在计算机科学、数

学、哲学等领域均有重要的意义。

参考文献

Cutland N. Computability, An Introduction to Recursive Function Theory. Cambridge: Cambridge University Press, 1980 (陈火旺 贵可荣)

kejisuanxing lilun

可计算性理论 (computability theory) 研究计算的一般性质的数学理论。计算的过程就是执行算法的过程。可计算理论的中心课题就是将算法这一直观概念精确化, 建立计算的数学模型, 研究哪些是可计算的, 哪些是不可计算的, 以此揭示计算的实质。由于计算是与算法联系在一起的, 因此, 可计算性理论也称算法理论。

直观上说, 求解一类问题的算法是一组规则。这组规则条数有限, 每一条都是可执行的 (可操作的), 并且这种操作性是绝对机械的, 即不论何人、何时对之进行操作, 只要输入数据相同, 其结果都是一样的 (惟一确定的)。作为算法的一组规则, 至少还应包含一条有关终止计算的条目。因此, 直观上算法所具备的特征为: 有限性、可执行性、机械性、确定性和终止性。求解一类问题的算法的执行过程就是从问题的初始数据开始, 依次执行每条规则, 直至终止。这个过程是有限的, 在数理逻辑中, 称这样的算法执行过程是一个“能行”过程, 并称可计算性理论为**能行性理论**。如果对终止性不加要求, 这类算法是不完全的, 或称是部分的, 即通常所说的过程。

例如, 求两个正整数 m, n (设 $m \geq n$) 最大公因子的欧几里得算法 (辗转相除法), 可用下述三条规划描述:

R_1 [求余数]: 以 n 除 m 得余数 r ;

R_2 [测试]: 若 $r = 0$, 终止计算, n 即为答案; 否则转到步骤 R_3 ;

R_3 [互换]: 变 m 的值为 n , 变 n 的值为 r , 转到步骤 R_1 。

依照这三条规则指示的步骤, 可在有限步里计算出任何两个正整数的最大公因子。计算的过程就是执行这三条规则的步骤所构成的序列。显然, 这个过程是确定的和有限的。

多少年来“算法”、“计算”这些概念似乎并不存在什么问题。到了 20 世纪 20 年代, 数学家们对此提出了疑问, 到底计算的实质是什么? 于是开始了对算法的概念精确化的研究。算法概念精确化的途

径很多,其中之一是形式地定义抽象计算机,把算法看作是抽象计算机的程序。进而把那些存在算法计算其值的“可计算问题”(或“可解问题”)精确定义为:能够在抽象计算机上编出程序计算其值的问题。在这种抽象计算机计算模型的基础上,就可以从理论上讨论哪些是可计算的,哪些是不可计算的。对于函数,凡存在抽象计算机程序计算其值者称为可计算函数。对于一般数学问题,凡存在抽象计算机程序求解者称为可解的问题。函数值的计算和问题的求解这两者是可以互相转化的。例如,在自然数论域里,询问问题“数 n 是否为素数”是否可解的,等于询问函数 $f(n)$

$$f(n) = \begin{cases} 1, & n \text{ 是素数} \\ 0, & n \text{ 不是素数} \end{cases}$$

是否可计算的。

可计算理论起源于对数学基础问题的研究。从 20 世纪 30 年代开始,为了讨论所有问题是否都有求解的算法,数学家和逻辑学家们从不同角度提出了几种不同的算法精确化定义。为简洁起见,许多数学家都起始于对自然数论域里的数论函数的可计算性的研究,提出了几种可计算函数定义。K. Godel, J. Herbrand 和 S. C. Kleene 于 1936 年定义了递归函数; A. Church 于 1935 年提出了 λ -转换演算; A. Turing 和 E. Post 分别于 1936 年和 1943 年提出了各自的抽象计算机模型(后人把 A. Turing 提出的抽象机称为图灵机); A. A. Mapkob 于 1951 年定义了正规算法; J. C. Shepherdson 于 1963 年定义无限寄存器机器; 50 年代末和 60 年代初,胡世华和 J. McCarthy 等人各自独立地提出了定义在字符串上的递归函数; 如此等等。后来陆续证明了,上述这些不同计算模型(算法精确化定义模式)的计算能力都是一样的,它们所刻画的函数类均相同,即它们是等价的。

可计算性理论的主要内容包括以下诸方面。

图灵机 一种在理论计算机科学中广泛采用的抽象计算机, A. Turing 于 1936 年提出,用于精确描述算法的特征。可用一个图灵机来计算其值的函数是可计算函数,找不到图灵机来计算其值的函数是不可计算函数。可以证明,存在一个图灵机 U , 它可以模拟任何其他图灵机。这样的图灵机 U 称为通用图灵机。通用图灵机正是后来的存储指令的通用数字计算机的理论原型。

λ 转换演算 一种定义函数的形式演算系统,是 A. Church 于 1935 年为精确定义可计算性而提出

的。他引入 λ 记号以明确区分函数和函数值,并把函数值的计算归结为按一定规则进行一系列转换,最后得到函数值。按照 λ 转换演算能够得到函数值的函数称为 λ 可定义函数。

丘奇-图灵论题 可计算性理论的基本论题,也称图灵论题。它规定了直观可计算函数的精确含义。丘奇论题说: λ 可定义函数类与直观可计算函数类相同。图灵论题说: 图灵机可计算函数类与直观可计算函数类相同。A. Turing 证明了图灵机可计算函数类与 λ 可定义函数类相同。这表明图灵论题和丘奇论题讲的是一回事,因此把它们统称为丘奇-图灵论题。直观可计算函数不是一个精确的数学概念,因此,丘奇-图灵论题是不能加以证明的。20 世纪 30 年代以来,人们提出了许多不同的计算模型来精确刻画可计算性,并且证明了这些模型都与图灵机等价。这表明图灵机和其他等价的模型确实合理地定义了可计算性,因此丘奇-图灵论题得到了计算机科学界和数学界的公认。

原始递归函数 自变量值和函数值都是自然数的函数,称为数论函数。原始递归函数是数论函数的一部分。首先规定少量显然直观可计算的函数为原始递归函数,它们是: 函数值恒等于 0 的零函数 C_0 , 函数值等于自变量值加 1 的后继函数 S , 函数值等于第 i 个自变量值的 n 元投影函数 $P_i^{(n)}$ 。然后规定,原始递归函数的合成仍是原始递归函数,可以由已知原始递归函数简单递归地计算出函数值的函数仍是原始递归函数。例如,和函数 $f(x, y) = x + y$ 可由原始递归函数 $P_1^{(1)}$ 和 S 递归地计算出函数值

$$\begin{cases} f(x, 0) = P_1^{(1)}(x) \\ f(x, S(y)) = S(f(x, y)) \end{cases}$$

比如, $f(4, 2)$ 可这样计算,首先算出 $f(4, 0) = P_1^{(1)}(4) = 4$, 然后计算 $f(4, 1) = S(f(4, 0)) = S(4) = 5$, 最后 $f(4, 2) = S(f(4, 1)) = S(5) = 6$ 。因此,和函数是原始递归函数。显然,一切原始递归函数都是直观可计算的。许多常用的处处有定义的函数都是原始递归函数,但并非一切直观可计算的、处处有定义的函数都是原始递归函数。

部分递归函数 为了包括所有的直观可计算函数,需要把原始递归函数类扩充为部分递归函数类。设 $g(x_1, \dots, x_n, z)$ 是原始递归函数,如果存在自然数 z 使 $g(x_1, \dots, x_n, z) = 0$, 就取 $f(x_1, \dots, x_n)$ 值为满足 $g(x_1, \dots, x_n, z) = 0$ 的最小的自然数 z ; 如果不存在使 $g(x_1, \dots, x_n, z) = 0$ 的自然数 z , 就称 $f(x_1, \dots, x_n)$ 无

定义。把如上定义的函数 f 加到原始递归函数类中,就得部分递归函数类。因为不能保证如上定义的 f 在一切点都有定义,故称其为部分函数。处处有定义的部分递归函数称为全递归函数。部分递归函数类与图灵机可计算函数类相同。对于每个 n 元部分递归函数 f ,可以编一个计算机程序 P ,以自然数 x_1, \dots, x_n 作为输入,若 $f(x_1, \dots, x_n)$ 有定义,则 P 执行终止并输出 $f(x_1, \dots, x_n)$ 的值,否则 P 不终止。

递归集 递归集最初是针对自然数的子集定义的,它们是存在算法判定每个自然数是否为其元素的集合。可以将递归集的概念推广到其他集合。所讨论的对象全体称为域,如果存在算法判定域中任意元素是否属于 A ,则称 A 为递归集。对于每个递归集,可以编一个计算机程序,以域中任意元素作为输入,计算机执行该程序都可给出适当的输出,表明该元素是否属于这个集合。有许多集合不是递归集。

递归可枚举集 如果对于集合 A 可以编一个程序 P ,输入域中任意元素 x ,若 $x \in A$,则 P 的执行将终止并输出“是”,否则 P 的执行不终止,就称 A 为递归可枚举集。 A 为递归可枚举集的充分必要条件是编一个程序枚举 A 的元素,即打印 A 的元素,使得对于 A 中任意元素,只要时间足够长总会在打印纸上出现。递归集都是递归可枚举集,但有些递归可枚举集不是递归集。有许多集合不是递归可枚举集。

可判定性和半可判定性 判定问题是无穷多个同类个别问题的总称。例如,2 是不是素数? 6 是不是素数? 这些都是个别问题,把这类个别问题概括起来,就得到一个判定问题:任意给定的正整数是不是素数? 这里的正整数集合称为该判定问题的域,给定域中的一个元素,判定问题就对应一个个别问题。对于一个判定问题,如果能够编出一个程序,以域中任意元素作为输入,执行该程序就能给出相应的个别问题的答案,就称该判定问题为可判定的(可解的)。例如,“任意正整数是不是素数”这个问题就是可判定的。对于集合 A ,域中任意元素是否属于它的问题称为集合 A 对应的判定问题。集合是递归集的充分必要条件为对应的判定问题是可判定的。因此,全体素数的集合是递归集。

对于一个判定问题,如果能够编出一个程序 P ,以域中任意元素作为输入,当相应的个别问题的解答是肯定的时候, P 的执行将终止并输出“是”,否则 P 的执行不终止,就称该判定问题为半可判定的

(半可解的)。可判定的问题总是半可判定的。集合是递归可枚举集的充分必要条件为对应的判定问题是半可判定的。

图灵在 1936 年证明,图灵机的停机问题是不可判定的,即不存在一个图灵机能够判定任意图灵机对于任意输入是否停机。图灵机的停机问题是半可判定的。图灵机的停机问题是很重要的,由它可以推出计算机科学、数学、逻辑学中的许多问题,如可以证明希尔伯特第 10 问题(刁番图方程是否有整数解的问题)是不可判定的。其证明也借助于停机问题的不可判定性。

可计算性理论是计算机科学的理论基础之一。早在 20 世纪 30 年代,图灵对存在通用图灵机的逻辑证明表明,制造出能编程序来作出任何计算的通用计算机是可能的,这影响了 40 年代出现的存储程序计算机(即冯·诺依曼型计算机)的设计思想。可计算性理论确定了哪些问题可能用计算机解决,哪些问题不可能用计算机解决。例如,图灵机的停机问题是不可判定的表明,不可能用一个程序来判定任意程序的执行是否终止,避免了人们为试图编制这样的判定程序而无谓地浪费精力。

可计算性理论中的基本思想、概念和方法,被广泛应用于计算机科学的各个领域。建立数学模型的方法在计算机科学中被广泛采用。递归的思想被用于程序设计,产生了递归过程和递归数据结构,也影响了计算机的体系结构。 λ 演算被用于研究程序设计语言的语义,例如,表处理语言就以 λ 转换演算为理论基础。

递归函数论的建立对于数学基础的研究具有十分重要的作用。为了证明不完全性定理, K. Godel 发明了原始递归函数。20 世纪初最伟大的数学家希尔伯特曾希望将整个数学形式化,建立了一个协调、完全的大系统。哥德尔不完全性定理表明,只要表达能力足够丰富,这种形式系统就不可能是完全的。例如,这种系统的协调性问题自身就无法在此系统内部得到证明。K. Godel 的发现对数学具有重要的影响,对认识论乃至整个哲学也有深刻的意义。

参考文献

1. Rogers H. Theory of Recursive Functions and Effective Computability. New York: McGraw-Hill, 1967
2. Cutland N. Computability, An Introduction to Recursive Function Theory. Cambridge: Cambridge University Press, 1980 (陈火旺 何自强 贵可荣)

kekaoxing sheji

可靠性设计 (reliability design) 研究计算机系统设计过程中,采取哪些措施来提高可靠性,使之达到可靠性指标要求的设计技术。可靠性设计是计算机工程设计的一个重要方面。在进行基本性能设计、经济性设计的同时,要进行可靠性设计。也就是从系统的总体设计、原材料、元器件选用、电路设计、结构工艺及可维性设计等各方面综合考虑,尽量挖掘各方面潜力,采取降额、冗余等各种措施来实现系统所要达到的可靠性指标要求。

可靠性是计算机系统在规定的条件下和规定的时间内,完成规定功能的能力。规定的条件通常包括环境条件、使用条件和维修条件,规定的时间是所需考虑的时间范围,规定的功能是指计算机的各项技术性能指标,能力是在规定的条件下和规定的时间内,完成规定功能的程度。可靠性可用可靠度或平均故障间隔时间定量表示。

可靠性设计一般遵循如下原则:

(1) 在满足功能和性能指标的前提下,把可靠性放在第一位。

(2) 在满足功能和性能要求的前提下,尽可能简化设计,采用简单的、少量的部件、器件、接口和电源等,以降低功耗、体积、重量和失效率,提高可靠性水平。

(3) 在减少数量的基础上尽量压缩品种、规格,以利于供应、质量监督、筛选测试和调试维修。

(4) 贯彻设计的标准化、系列化和模块化原则,以利于系统扩充和各型号间的兼容性。

(5) 对性能、可靠性、成本等要综合考虑。

可靠性设计内容包括:①建立系统及各分系统的可靠性模型;②可靠性预计;③可靠性指标分配。可靠性设计通常不是一次就可全部完成,要经过预计、分配、再预计、再分配的反复过程;而且在研制阶段基本结束后,还要经过各种可靠性试验,对所暴露出的缺陷采取必要的措施,使可靠性进一步提高,最后才能将可靠性设计定型。

可靠性模型 以产品的功能框图、原理图和工程图为基础,建立系统、分系统或设备的可靠性框图,即产品的可靠性物理模型。在框图中方框表示评定系统可靠性时必须考虑的单元,所有连接方框的线不考虑可靠性值,即可靠性等于1。产品各单元间连接用的导线或连接器的可靠性必须考虑时,可单独放入一个方框或作为某个单元的一部分。所有方框的失效率是互相独立的,与它们的排列顺序无关。

找出各方框的可靠性数学表达式及各方框之间的可靠性关系,建立以各方框的可靠性为基础的系统、分系统或设备的可靠性表达式,即可靠性模型。其目的是确定产品成功或失败的概率与各单元成功或失败概率的关系。

以某计算机为例,它是由中央处理器(CPU)板、内存板和输入输出(I/O)板及电源和机箱5个部分组成的无冗余简单串联系统,图1是其可靠性框图。

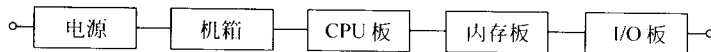


图1 某计算机可靠性框图

若每个部件的可靠性为 R_i , 则总的可靠性 R_T 为

$$R_T = \prod_{i=1}^5 R_i$$

各部件的失效率 λ_i 是个常数, 可靠性 $R_i = \exp(-\lambda_i t)$, 则总可靠性为

$$R_T = \prod_{i=1}^5 R_i = \exp\left(-t \sum_{i=1}^5 \lambda_i\right) = \exp(-\lambda_T t)$$

式中 t ——工作时间;

λ_T ——产品所有部件失效率的总和。

可靠性预计 其目的是根据可靠性框图和数学模型预计计算机系统、分系统或设备的可靠性,并确定所提供的设计方案是否达到了任务书或合同规定的可靠性要求,为确定设计方案和选择元器件、材料

等提供依据。可靠性预计工作的基本要点是:①应对计算机系统、分系统或设备做出可靠性预计,为寿命周期费用、后勤保障分析、产品的应用效能分析提供依据。特别要根据预计中的数据和结果指出失效率较高的设备,为设计修改、故障分析提供依据。②在方案构思阶段和正式设计阶段采用不同的方法进行可靠性预计。③随着设计、元器件的选择和使用环境的改变,可靠性预计工作应及时修改,使之能反映产品的真实水平。

在方案构思阶段的可靠性预计是用比较有限的资料较迅速地做出可靠性预计,其目的是对方案的合理性、可行性作出概略的估计与判断。此种预计法无须进行详细的分析和精确的计算,通常采用元

器件计数法。该方法需要三方面数据:①设备所有元器件的种类及数量;②拟采用的各种元器件的质量等级;③设备的使用环境。

正式设计阶段的可靠性预计通常采用元器件应力分析法,这种方法需要大量的详细信息,尤其是元器件各种应力在内的详细数据,其设计精度比较高,可以满足工程设计的精确性要求。

有关元器件计数预计法和元器件应力分析预计法的详细情况参见 GJB 299—87《电子设备可靠性预计手册》。

可靠性指标分配 从系统或整机的角度出发,根据经过论证确定的系统可靠性指标要求,按照一定的原则和方法分配给各分系统,进而再将各分系统的可靠性指标分配给各个部件、电路,直至元器件、印制板、接插件等。可靠性指标分配要确保分配的合理性(协调性)、技术上的可行性和资源利用的经济性。可靠性指标分配的方法有许多种,具体采用哪一种,取决于所构成的系统模型,已掌握的数据材料,现有技术条件以及体积、重量、功耗等限制条件。可靠性指标分配应反复进行,当某一分系统所分配的指标技术上不可行时,应修改方案,进行重新分配,直至满足设计要求为止。主要的可靠性指标分配方法有:考虑重要程度和复杂程度的代数分配法,考虑复杂程度的分配法,各分系统失效概率可预计情况下的分配方法,顺序分配法,按经验比率分配法等。

可靠性设计软件 可靠性设计是一项复杂的系统工程,为了科学、准确、高效地做好此项工作,出现了可靠性设计软件,利用计算机,在电子设计等软件的配合下进行可靠性设计。这类软件可进行可靠性预计分析、可靠性框图分析、失效数据分析、故障模式、影响及危害度分析、故障树分析、寿命周期费用分析,发现设计中的薄弱环节,为设计改进或生产过程控制提供依据。

参考文献

1. 傅佩琛,赵霖,张军英. 计算机系统硬件软件可靠性理论及其应用. 北京:国防工业出版社,1990
2. Siewiorek D P, Swarz R S 著. 可靠系统的设计理论和实践. 袁由光,曹泽翰,刘志模等译. 北京:科学出版社,1988 (刘恩德)

ke kuochong yuyan

可扩充语言 (extensible language) 具有扩充机制的程序设计语言。一个程序设计语言称为可

扩充的是指用户能根据自己的需要,利用该语言提供的扩充手段,往该语言中增加新的成分或新的功能,而又无需改动该语言的编译程序。

扩充机制就性质分,有词法扩充、语法扩充和语义扩充等层次。例如,关键词和运算符换名,通过宏展开引进新的文字等属于词法扩充。定义复合数据类型,引进新的运算符等属于语法扩充。确定新运算符的运算规则(如结合律、交换律)和原有运算符的重载(如改变其定义域)等属于语义扩充。

历史上把可扩充性作为特色加以强调的语言有 ELI(相应的实现系统为 ECL), ALGOL 68, FORTH, SIMULA 67 等。其中 SIMULA 67 可以在系统类的基础上插入新的类说明,从而把它扩充为各种专用语言。这种扩充方式一直沿用至今。

可扩充语言的研究兴起于 20 世纪 60 年代中期。当时,日益扩大的计算机应用领域向计算机语言提出了各种功能要求。如果把这些功能都纳入一个计算机语言中便会使语言的规模过度膨胀。由此人们企图以适度规模的语言加上可扩充功能来适应这种需要。美国哈佛大学的 T. E. Cheatham 等人是研究可扩充语言的带头人。该领域在 70 年代初成为研究热点,到 70 年代中期,便开始降温。著名程序设计语言评论家 J. Sammet 在 1981 年指出:“可扩充语言作为研究方向已经死亡”。其原因是:程序设计语言的研究在 70 年代经历了一场从追求功能丰富到追求程序可靠的变革。此外,可扩充语言的研究并未像人们预期的那样给程序设计带来革命性的变化。但是,有关的研究成果仍然体现在当代的一些重要程序设计语言中。 (陆汝铃)

kekuozhan zhibiao yuyan

可扩展置标语言 (extensible markup language, XML) 一种具有可扩展性、结构化并可以提供有效性检查的置标语言。它提供了文档内嵌入标记的描述标准,并克服了超文本置标语言 (HTML) 在标记集合上缺乏扩展性的局限性。可扩展置标语言 (XML) 将内容结构、表现样式以及超链接三类信息分开表示,克服了 HTML 网页信息中这三类信息不易区别的缺陷。

利用 XML 语言可以为各个应用领域建立各自的置标语言。目前已经制定了基于 XML 的实用置标语言有:化学置标语言 (CML)、数学置标语言 (MathML)、图形学置标语言 (SVG) 和电子书 (Doc-Book) 等。万维网联盟 (W3C) 国际组织为 XML 文

档的语法结构制定的 XML Schema 语言也是由 XML 定义的。

以下特点使 XML 成为置标语言的开放标准。

结构 XML 文档可以在它的文档类型说明中说明使用某个用 XML 语言表示的文档类型定义 (DTD), 用以规定文档中出现的标记和元素结构的语法约束。

样式 XML 文档的显示格式由样式表确定, 适用于描述 XML 文档样式表的语言有 XSL 和 CSS2。它们也是由 XML 语言定义的。XSL 可用来将 XML 文档转化为结构良好的 HTML 文档。

超链接 XML 文档的超链接由 XML 链接语言 (XLink) 提供, 它是一种用 XML 元素向 XML 文档中加入了比 HTML 更加灵活的链接机制, 不仅支持 HTML 的单向链接, 还支持多目的和多方向链接。它还允许把链接单独提出来存放在数据库中或者单独的文档中。链接到 XML 文档内部的目标定位使用 XPointer 规定的格式。XLink 本身也是由 XML 定义的一个具体的置标语言。

Unicode 多语编码 XML 采用 Unicode 字符编码系统 (参见汉字编码字符集), 从而支持世界上几乎所有的主要语言, 并且不同语言的文本可以在同一文档中混合使用。

面向可扩展置标语言 (XML) 的应用编程技术主要有文档对象模型 (DOM) 和 XML 的简单应用程序编程接口 (SAX)。DOM 是万维网联盟 (W3C) 制定的一种独立于平台和编程语言的应用编程接口规范。SAX 是一个事件驱动型的应用编程接口, 在原理上比 DOM 简单。

参考文献

1. W3C. Extensible Markup Language (XML) 1.0 (Second Edition). October, 2000 (<http://www.w3.org/TR/2000/REC-xml-20001006>.)
2. Bosak J and Bray T. XML and the Second-Generation Web. Scientific American, 1999; 89 ~ 93
3. The Unicode Consortium. The Unicode Standard, Version 3.0., February 2000. <http://www.unicode.org/unicode/standard/versions/Unicode3.0.html> (瞿裕忠)

ke panding wenti

可判定问题 (decidable problem) 具有求解算法的问题。判定问题的一系列研究结果是 20 世纪最重大的学术成就之一。Hilbert 1900 年在国际

数学家大会上提出的 23 个问题中的第 10 个问题, 即丢番图方程解的存在性判定问题就是一个著名的判定问题。另一个著名的判定问题是所谓的数理逻辑基本问题: 在一阶逻辑中, 是否存在判别任意命题为定理的一般方法。正是对这些问题的研究, 直接推动了递归论的产生。在 1936 年, S. Kleene 提出了递归函数的概念, A. Turing 提出了图灵机模型, A. Church 和 A. Turing 独立地证明了数理逻辑的基本问题是不可判定的。

由于各种形式的判定问题都可转化成某一集合 A 的从属关系的判定问题: 是否存在算法能判定任意 x 是否属于集合 A 。因此, 可判定问题可用下述形式定义: 若 A 是递归集, 则称 A 的从属关系的判定问题是可判定的, 或称递归可解的。

由于绝大多数著名的判定问题都是不可判定的, 所以通常研究一个问题的可判定性一般是指证明它的不可判定性。

参考文献

1. Rogers H. Theory of Recursive Functions and Effective Computability. New York: McGraw-Hill 1967
2. 莫绍揆. 递归论. 北京: 科学出版社, 1987
(马绍汉 李大兴)

keshi biancheng yuyan

可视编程语言 (visual programming language) 一类用图形符号描述计算任务的处理对象和处理过程的语言。这种语言与传统程序语言的最大区别是: 传统语言是由正文形式表示的一维字符串结构, 而可视语言则是由图形符号的空间排列所表示的多维结构。

现实生活中, 图形和图表往往比文字更能形象地表达人们的思想意图, 也更有助于思想的了解和交流。由此早在 20 世纪 50 年代计算机科学中就引入了描述计算过程的逻辑图概念, 以后又把它逐步规范化为流程图, 用它们描述算法, 作为进行程序设计的辅助工具。60 年代它就成了最早的可视语言, 1966 年 W. R. Sutherland 第一个用逻辑图作为可视语言表示程序并给予解释性实现, 开创了可视语言的先例。1969 年 T. O. Ellis 等人建立的 Grail 首次用流程图表示计算步骤并直接编译成执行程序。这可能是可视语言的最早编译实现, 其中流程图框中内容是用机器语言陈述的。以后又建立了很多流程图和流程图变形 (如 N-S 图) 式的可视语言。非流程图式可视语言的最早成果大约是 C. Christensen

的 AMBIT/G(1968 年)和 AMBIT/L(1971 年),它把程序和数据都表示成有向图。程序用模式匹配进行操作,相当复杂的算法(如存储碎片收集)都可以生动地描述成图上的局部变换。第三类可视语言称为 HiGraphs,是 D. Harel 在 1988 年建立的,它允许结点中包含其他结点,它限制产生专用的可视语言,例如 Miro(1988 年)就是一种定义操作系统安全约束的 HiGraphs 语言,StateMaster(1989 年)则是一种计算机用户界面的 HiGraphs 语言。第四类可视语言如 M. Zloof 等人建立的 QBE(1977 年),它允许用户用二维表格指出在关系数据库上的查询。1981 年又把这种思想扩充到办公自动化领域建立 OBE。第五类是 Petri 网式语言。1981 年 T. Ae 等建立的 MOPS-2 用带色 Petri 网按可视方式构造和模拟并发系统。VERDI(1987 年)是一个 Petri 网式可视语言的解释性系统。用户在这种程序执行时可以看到令牌在网上移动的程序动画。第六类是数据流图式可视语言,它们以 PROGRAPH(1983 年)和 Lab-VIEW 为代表。第七类是自动生成用户界面的可视语言,例如 Peridot(1988 年)和 UIMS(1989 年)。此外,还有一些其他类型的可视语言。

这些可视语言都体现了用图形来进行编程的特点。其图形符号大都是软件工作者所熟悉的。近几年对其他(如商业营销、企业管理)领域的对象图形和操作符号也进行了一些讨论,可以建立针对这些领域工作者所熟悉的图形符号的可视语言。

G. TorTora 提出一种可视语言的形式描述:可视语言程序由基本图符的空间排列组成。基本图符分为基本对象图符和对这些对象的处理图符。可视语言的终结符号集包括基本图符和空间操作符两部分。最基本的空间操作符至少包含横向连接、纵向连接和空间覆盖等。一个可视语言表示成三元组 (ID, Go, B) ,其中 ID 是基本图符字典,它刻画了每个基本图符的逻辑部分(含义),物理部分(显示图形)和类型(对象或处理)。 Go 是上下文无关文法,它指出如何用空间排列基本图符构造合法的复合图符。每个图符,不论是基本的还是复合的,都具有逻辑部分和物理部分的对偶表示。一个可视程序也是一个复合图符。 B 是知识库,它包含了为构造一个已给可视程序的含义所需的具体领域的信息。 (ID, Go, B) 构成一个可视属性文法 VAG,它以 Go 为基础。VAG 的语义规则是与 Go 中的产生式相关联的,它指出已给派生树如何合成可视程序的含义。实际的语义规则由包含在 B 中的信息以具体例子

说明每个规则的未定部分。

可视语言的实现要求为它建立一个实现环境,称为可视程序开发环境。图符编辑程序负责编辑可视程序 P 的图符,经模式分析程序把空间结构的图形程序变换成模式串,语法分析程序由模式串产生 P 的分析树,再经语义映射程序得到 P 含义(程序的机内表示),再通过解释程序执行或经编译程序产生可执行代码。为了支持大型系统,环境中还应该数据库、视图工具、浏览程序、项目管理程序等工具。如 1990 年 M. Hirakawa 等人建立的 HiVisual 可视程序环境就支持对程序开发和系统操作的导航,对基于面向对象概念的可视程序和系统操作提供解释机制,提供程序的自顶向下和自底向上开发,提供对现有系统的集成。

可视语言现在仍处于实验研究阶段,要真正走向实用还有相当距离。可视语言以问题空间的处理对象为单位定义基本图符。在某种意义上说它也是面向对象的。而且把对象图像化更易于人们的使用和理解。所以它是很有发展前途的。

参考文献

Ichikawa T, Jungert E, et al. (ed.) Visual Languages and Applications. New York: Plenum Press, 1990

(李万学)

keshi chengxu sheji

可视程序设计 (visual programming) 用可视语言编写可视程序的方法与过程。

在现实生活中用户所见到的绝大多数对象都是多维的,传统的程序设计要求把这种多维的对象强行变为一维的符号串描述才能被计算机所接受。这就好比计算机是瞎子,只能听用户通过口述把多维对象描述成规定格式的一维文字一样,给用户增加了很大的负担。如果用户能直接把这些多维对象送入计算机,并在计算机中规定了直接对这些多维对象的处理操作,将给用户使用计算机带来很大的方便。现有的一些可视程序设计系统表明,非程序员用户经过短期培训就可以用可视语言编写出相当复杂的程序,这是因为可视语言确实更适合人们的习惯。目前已经建立了大量的可视程序系统,几乎把软件工作者用过的所有辅助编程的图形(如流程图、数据流图, Petri 网图,有向图等等)以及这些图形的变种都作为模型建立了各种可视语言。直接用这些基本图形符号,制定一套对这些图符的处理操作,再规定一套语言的语法和语义规则就形成了一

个完整的可视语言。用这种语言编写的可视程序也是一个图形。

可视程序设计不包括用传统正文语言定义图像的系统,如 Macintosh Toolbox 或 X-11 Window Manager Toolkit 等,也不包括绘图软件包的系统,如 Apple Macintosh MacDraw,因为它们并不是用图符来产生程序。

用可视语言编制可视程序应该在一个可视程序设计开发环境下进行。用户首先用它熟悉的图符结合其逻辑含义构思一个解题的图形,如果有图形输入设备,用户可将构图画在纸上用图形扫描器输入,没有图形输入设备,对确定的可视语言建立一个可视程序编译程序,这个编译程序的用户界面中列出了所有基本图符的图形作为选单的内容,用户编写可视程序时可在屏幕上通过鼠标指点选单上的图形拼凑成解题的可视程序以后就等待计算机去执行这个程序。

在执行过程中,根据屏幕提示,输入相应的数据,用户使用起来非常方便。当然,要建立一个这种可视程序开发环境系统的任务是繁重的,除了建立上述编辑程序之外,还要对基本图符给出逻辑含义。在编辑图形的同时或以后,系统根据可视语言的语法和语义导出拼凑的可视程序的逻辑含义(即计算任务和计算步骤),自动解释执行这个可视程序或通过编译程序自动产生该可视程序的执行代码。如果定义的基本图符所表示的对象的图形屏幕形像和逻辑含义都符合用户习惯,用户用可视语言开发自己的软件就相当容易。所以可视语言逐渐向行业(或专业)靠近,变成行业(或专业)专家的语言将成为发展方向。

目前可视程序设计系统基本上仍是一些小型的实验性系统,离实用化的目标尚远,尤其是对复杂图形的(大)可视程序,如何使图形分层,降低复杂程度,达到使其结构清晰、易读等都是需要努力的。

参考文献

1. CHANG S-K. (ed.) Principles of Visual Programming Systems. Prentice Hall Inc., 1990
2. CHANG S-K. (ed.) Visual Language and Visual Programming. New York: Plenum Press, 1990

(李万学)

keshi dianhua

可视电话 (videophone) 使通信双方在进行通话时可以互相看到对方图像的一种电话系统。可视

电话在传输信道上可分为 PSTN 网(公用电话网)、ISDN 网(综合业务数字网)、IP 网(如互联网)和专用网等多种方式。由于传输带宽的限制,可视电话一般使用较小的屏幕和较低的视频帧率。在 PSTN 上工作的可视电话,每秒钟可以传输 10 ~ 15 帧画面;在 ISDN 上工作的可视电话,每秒钟可以传输 15 帧以上的画面。1996 年国际电信联盟公布了 PSTN 多媒体可视电话国际标准 ITU-H. 324 标准。

目前,可视电话产品主要有两种类型,一类是以个人电脑为核心的可视电话,除电脑外还配置有摄像机(或小型摄像头)、麦克风和扬声器等输入输出设备;另一类是专用可视电话设备(如一体型可视电话机),它能像普通电话一样,直接接入家用电话线进行可视通话。由于普通电话线普及率很高,因此,随着社会对电话通信多样化需求的增长,电话将从单一的话音通信发展为音视频(及数据)一体的多媒体通信形式。在公用电话网上工作的可视电话最具发展潜力,可视电话作为一种新型的通信方式将进入千家万户。

可视电话不仅适用于家庭生活,而且还可以广泛应用于各项商务活动、远程教学、保密监控、医院护理、医疗诊断、科学考察等多种不同行业和领域。

(钟玉琢)

kexin jisuanji xitong pinggu zhunze

可信计算机系统评估准则 (trusted computer system evaluation criteria, TCSEC) 美国国家计算机安全中心(NCSC)于 1983 年制订的计算机系统安全等级划分的基本准则。通常称它为“橘皮书”,共分为四类七级: D、C1、C2、B1、B2、B3、A1。

D 级,安全保护欠缺级 凡经检测,安全性能达不到 C1 级的即被定为 D 级。D 级并非没有安全保护功能,只是太弱。

C1 级,自主安全保护级 通过可信计算基定义和控制系统中命名用户对命名客体的访问。其实施机制(例如,访问控制表(参见访问控制))允许命名用户和(或)以用户组的身份定义并控制客体的共享,并阻止非授权用户读取敏感信息。

可信计算基(TCB)是指为实现计算机处理系统安全保护策略的各种安全保护机制的集合。

C2 级,受控存取保护级 与自主安全保护级相比,本级的可信计算基实施了粒度更细的自主访问控制。它通过登录规程、审计安全性相关事件以及隔离资源,使用户能对自己的行为负责。

B1 级,标记安全保护级 本级的可信计算基具有受控存取保护级的所有功能。此外,还提供有关安全策略模型、数据标记以及主体对客体强制访问控制的非形式化描述;具有准确地标记输出信息的能力;消除通过测试发现的任何错误。

B2 级,结构化保护级 本级的可信计算基建立在一个明确定义的形式化安全策略模型之上。它要求将 B1 级系统中的自主和强制访问控制扩展到所有主体与客体。此外,还要考虑隐蔽通道。本级的可信计算基必须实行结构化处理,成为关键保护元素和非关键保护元素。可信计算基的接口也必须明确定义,使其设计与实现能经受更充分的测试和更完整的复审。B2 级加强了鉴别机制;支持系统管理员和操作员的职能;提供可信设施管理;增强了配置管理控制。B2 级系统具有相当的抗渗透能力。

B3 级,安全域级 本级的可信计算基满足访问监控器需求。访问监控器是指监控主体和客体之间授权访问关系的部件,负责仲裁主体对客体的全部访问。访问监控器本身应该是抗篡改的,必须足够小,能够分析和测试。为了满足访问监控器需求,可信计算基在其构造时就应排除对实施安全策略来说并非必要的代码;在设计和实现时,从系统工程角度将其复杂性降低到最小程度;B3 级支持安全管理员职能,扩充了审计机制,当发生与安全相关的事件时能发出信号,并提供系统恢复机制。B3 级的系统具有很高的抗渗透能力。

A1 级,验证设计级 本级的安全功能与 B3 级相同,最明显的不同是本级必须对相同的设计运用数学形式化证明方法加以验证,以证明其安全功能的正确性。本级还规定了将安全计算机系统运送到现场安装所必须遵守的程序。

(胡道元)

kehu-fuwuqi jisuan

客户-服务器计算 (client /server computing) 把复杂任务从功能上分割成几个不同的部分,再分配到具有前后端结构的分布式计算环境中,在前端客户机上运行应用程序,而后端服务器则提供某些特定服务的一种计算模式。服务器提供的服务有数据库服务、文件服务和通信服务等。客户-服务器计算环境的基本组成如图 1 所示。客户-服务器计算环境通常由以下几个部分组成:①通信网络,可以是局域网或广域网;②服务器,可以是个人计算机、工作站、小型计算机、大型计算机等;③客户机,一般是个人计算机;④应用程序开发界面,例

如 SQL 数据库语言等;⑤图形用户界面,例如 Windows, Motif 等。

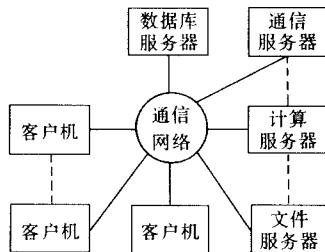


图 1 客户-服务器计算环境的基本构成

客户-服务器计算环境起源于 20 世纪 80 年代中期,它不同于当时的个人计算机的单一式处理方式,也不同于小型计算机与大中型计算机的宿主式计算方式以及计算机网络的点对点计算方式,而是一种能够通过通信来共享位于不同地域的计算机服务器的软、硬件资源的计算模式。

支持客户-服务器计算的主要技术有 3 种。即远程过程调用、分布式数据库管理系统和通信协议。

(1) 远程过程调用 指用户可以像调用本地过程一样调用不同地域的不同计算机上的过程,从而使使得应用程序设计人员不必设计和开发有关发送和接受信息的实现细节。实现远程过程调用必须解决好下述 4 个问题:①存根子程序设计。存根子程序帮助用户掩盖远程过程调用的细节,使用户像调用本地过程那样调用远程过程。②绑定问题。绑定程序使用户程序和特定的服务程序发生联系,也就是正确定位用户程序所需要的服务程序,从而使远程过程调用成为可能。③可靠性问题。④参数传递问题。即在进行远程过程调用时,如何在不同机器之间正确有效地传递参数。

(2) 分布式数据库管理系统 将分散在不同结点上的数据资源通过通信网络连接起来,统一进行管理和控制,使其在逻辑上形成一个统一的数据库系统(参见分布式数据库系统)。

(3) 通信协议 为远程过程调用和分布式数据库管理系统提供通信规则,从而使使得客户-服务器计算成为可能(参见网络协议)。

客户-服务器计算环境具有如下优点:

(1) 运行性能高 在分散处理的应用环境中,由于客户-服务器计算把处理功能进行了适当的划分,显著地减少了网络上的流通负担,提高了整个系统的运行性能。由于服务器端不需支持应用功能,

使服务器端的内存和中央处理器可全部供数据库管理系统等服务器中的后台程序使用。客户机端主要执行应用程序,不需提供数据管理功能,因此,客户机端的内存和中央处理器可全部提供给应用程序使用。另外,数据库服务器提供多种不同层次的锁定和快速复制功能,从而提供了更好的并行控制以确保多个用户在同一时间内存取相同的数据,这进一步提高了运行性能。

(2) 便于最终用户使用 在客户-服务器计算环境中,客户机可根据最终用户需要装载文档处理软件、决策支持工具、前端电子邮件、数据库请求程序以及图形用户界面等,从而客户机为最终用户提供比宿主计算环境更为方便的用户界面。

(3) 具有多功能的应用程序开发接口 客户-服务器计算环境提供由客户机用户使用的标准语言(例如 SQL)和多种开发支持工具,这些标准语言和支持工具构成方便的多功能应用程序开发接口。用户可以使用这些应用程序开发接口透明地请求服务器的各种有效服务。

(4) 开放的体系结构 客户-服务器计算环境具有开放的体系结构,它们大都使用国际流行协议 TCP/IP 或 SPX/IPX,从而使得支持这些协议的不同厂家的软、硬件产品可以集中到同一客户-服务器计算环境中。

(5) 良好的可扩充性 由于客户-服务器计算环境具有开放的体系结构以及应用程序和服务程序分别在客户机和服务器上执行,客户-服务器计算环境具有良好的可扩充性。可以在后端服务器功能不变的前提下,对服务器硬件或软件进行扩充或更新换代,这种更新换代不会给前端应用程序带来负面影响。同样,在后端服务器负载能力允许的条件下,可以自由增加客户机的台数。客户-服务器计算环境的可扩充性不仅使用户在应用软件方面的投资得到保障,并为用户提供一种低消耗的逐步更换设备的途径,从而使得用该技术构筑的信息系统能方便迅速地跟上新技术的变化。

建立客户-服务器计算环境的主要难点在于:

(1) 安全性问题。客户-服务器计算环境将应用程序和服务信息分布在整个计算环境中,且允许用户访问外界信息或允许外界用户访问计算环境中的信息,因此,必须设置相应的访问权限、口令、密码或其他保密措施。

(2) 故障隔离问题。

(3) 数据的完整性和一致性问题。

(4) 正确划分应用程序,以获得最佳系统性能问题。

(5) 应用程序开发人员、使用和管理人员的教育与培训问题。

(6) 系统维护问题。

客户-服务器计算模式已日益普及,随着网络技术的成熟,越来越多的用户将采用这一模式设置计算机系统,但有些用户从降低维修成本、保证系统安全性等角度出发,仍会采用宿主-终端计算模式。

(张尧学)

kehu - fuwuqi moshi

客户-服务器模式 (client/server mode) 参见客户-服务器计算。

kejian

课件 (courseware) 根据教学目标进行设计并反映某种教学策略的,用于传递教学内容、实施课程教学活动的一种计算机软件。

课件是计算机辅助教学(CAI)系统的核心,其质量是实施 CAI 的关键。从教学角度分析,课件由教学内容(学科知识)、教学策略(教学过程与方法)和学生模型(学习者的信息)3 部分组成。课件的设计,应从这 3 方面着手。不同的教学内容应采用不同的教学策略,不同的策略适用于不同的教材,不同的教材、不同的教学策略适合于不同认知能力的学生。

从计算机软件角度分析,课件包含数据(与教学内容相关的教学信息、教材、参考资料等)、控制(用于对学习过程进行引导、处理、诊断和评价,决定教学过程如何进行)及人机接口(确定学习者与课件的交互方式以提高教学效果)3 个部分,三者协调完成软件运行及教学任务。

课件可以分成许多不同的类型:

(1) 根据采用的教学模式可分为:①示教型(用于演示抽象、复杂的现象和过程,适合课堂教学使用);②模拟实验型(仿真电子电路实验、化学实验等,效果较好,可节省实验费用);③授课型(适用于自学某些课程,对职业培训和成人教育较为合适);④训练自测型(可提高外语、医学、程序设计等课程的课外学习效率);⑤游戏型(计算机生成一种带竞争性的学习环境,通过游戏达到预定的教学目标,它把趣味性、教育性、科学性融为一体,做到“寓

教于乐”);⑥咨询型(学习者可主动提出各种问题,要求课件给予回答和说明,有利于学生自主学习)。此外,还有问题求解型、发现学习型等。

(2) 根据教学信息的呈现方式可分为:①框面型(以固定框面的形式呈现教学信息);②自动生成型(由程序自动生成教学信息);③数据库型(教学信息是从教材库中提取并呈现给学习者);④超文本型(各知识点作为节点,通过超链接相互连接,组成一个知识网)等。

(3) 根据课件的使用对象可分为:①助学型(学习者是课件的主要使用者,此类课件要具有完整的知识结构,能反映一定的教学过程和教学策略,有友好的人机界面,学生可以在个性化的教学环境下进行自主学习);②助教型(教师是课件的主要使用者,它能辅助教师更好地完成课堂教学任务);③教学结合型(兼顾教师与学生两者使用的课件)。

(4) 根据课件的使用环境可分为:①单机版(以 CD 光盘作为载体,可在个人计算机上独立运行);②网络版(以局域网或 Internet 网为依托,将网络作为传输教学内容的载体,为实现地理上分散的教学活动而开发的课件)。

无论哪一种课件,它们的开发过程都包括教学需求和教学目标的论证,课程的教学设计,课程的结构设计(教学单元控制设计和内容设计),课件的制作,测试和评价,运行维护。每个教学单元的设计与制作要求做到:①内容部分要重点突出、意义明确、生动有趣和布局合理;②问题部分的设计要包括提问、学生答案的输入、判断比较和反馈信息等;③控制设计要清晰、直观、合理,能适应多种不同学习者的需求。一个优秀课件的开发往往需要教育专家、任课老师和计算机专家等几方面人员的参与,许多情况下,还需要听取学生的意见。

课件的开发周期长、成本高、难度大,而且需要有相关领域的专门知识,从事这方面工作的人员一般都是教育领域的专家和教学经验丰富的老师,他们往往并不具备计算机编程的能力。为了解决这一问题,一类称作“著作工具”的软件应运而生。

课件著作工具的主要特点是简洁易用,它们不需要或者很少需要用户具有程序设计语言、数据结构和算法等方面的知识。用户通常在几天甚至几个小时之后,就可以学会使用它们进行课件的开发工作。课件著作工具往往都提供一些样板应用,对这些样板进行适当修改即可快速生成一个新的课件。

为了克服课件著作工具灵活性较差的缺点,许

多著作工具都能支持一种或几种脚本语言,例如 VBScript, JavaScript 等,它们简单易懂,容易掌握。使用脚本语言编写的程序可以嵌在课件中,从而方便地实现用户定制的功能。

参考文献

William Horton. Designing Web-Based Training: How to Teach Anyone Anything Anywhere Anytime. John Wiley & Sons, Inc., 2003 (张福炎)

kongjian fuzaxing

空间复杂性 (space complexity) 解答问题时以空间度量的算法复杂性。算法的空间需求是指计算所需要的计算机存储量。在图灵机计算模型中,使用的空间量定义为读写头访问的不同的带方格数。因为访问的带方格数不可能多于计算步骤,所以任何在 $T(n)$ 时间内可解的问题也必可在 $T(n)$ 空间内可解。更精确地说,输入本身占有的空间是不应该计算在空间用量内的,所以图灵机的带分成输入带和工作带,空间复杂性是指图灵机运行时带头读写工作带上经过的带方格数。如果是随机存储器模型,空间复杂性是指工作单元数(有时按工作单元的二进制数)。大量实例表明,空间复杂性与时间复杂性一样,也是一个与计算模型无关的概念。

在多项式时间内可解的所有问题都可以在多项式空间内解决,但是否存在在多项式空间内可解的问题不能在多项式时间内解决,这仍然是一个未解决的问题。一般认为有这样的问题存在。此即著名的 $P \neq PSPACE$ 问题。

用图灵机计算模型,可以得到很简单的带压缩定理:如果语言 L 能够被图灵机 M 在空间 $s(n)$ 内接受, c 是任意 $(0, 1)$ 之间的实数,则 L 可以被另一图灵机 M' 在空间 $cs(n)$ 内接受。

这表明在空间复杂性函数中,常系数是一个次要的因素。人们更为关心的是空间复杂性在输入规模趋于 ∞ 时的渐近性态,即它的增长率。

还有一个与时间复杂性截然不同的定理是:如果语言 L 能够被不确定图灵机 M 在空间 $s(n)$ 内接受,则 L 可以被另一确定图灵机 M' 在空间 $s^2(n)$ 内接受。

并行算法的空间复杂性和时间复杂性有相互折算的关系,空间耗费大,即并行程度高,则并行算法的执行时间可以降低,但也不是说空间用量任意地增加,并行时间复杂性可以任意地小,故权衡并行算法的时空折衷关系,设计好的空间体系结构(如

n -维立方体,星形图等)是并行算法空间复杂性研究主要关心的问题。

参考文献

1. Hopcroft J E, Ullman 著. 自动机理论、语言和计算. 徐美瑞译. 北京: 科学出版社, 1986
2. Balcázar J L, Díaz J, Gabarró J. Structural Complexity, I, II. Springer-Verlag, 1988
3. Akl S G. The Design and Analysis of Parallel Algorithms. N. J.: Prentice Hall (朱洪)

kongjian shujuk

空间数据库 (spatial database) 以描述、存储和处理查询空间数据为特色的数据库。与其他类型的数据库相比,空间数据库在对传统数据库类型的处理能力的基础上增加了对空间数据的处理能力。它从数据模型、查询语言、查询处理和存储方法诸方面对空间数据提供了全面的支持,是当前地理信息系统 (GIS) 和计算机辅助设计 (CAD) 等应用的基础工具。

空间数据是用于表示空间物体的位置、形状、大小和分布特征等方面信息的数据,用于描述二维、三维和 multidimensional 分布的关于区域的现象。空间数据的特点是不包括物体本身的空间位置及状态信息,还包括表示物体的空间关系(即拓扑关系)的信息。属性数据为非空间数据,用于描述空间物体的性质,对空间物体进行语义定义。

空间数据库除了要提供对空间数据存储与访问功能外还要提供对常规数据的访问能力,所以大多数空间数据库是以现有成熟的数据库管理系统为基础建立的。其系统结构如图 1 所示。

上层是各种空间应用基理(逻辑),如代表 GIS 应用、CAD 应用等;中间是空间数据库系统,它结合传统的数据库技术实现对空间对象的存储与查询,并提供对空间应用开发的支持;下层是常规的数据库管理系统,一般常见的是对象-关系数据库管理系统和面向对象数据库管理系统,实现对常规数据的存储和查询。

空间数据库部分也分为 3 层,其中空间应用接口层提供应用的访问接口,包括抽象数据类型、数据模型、形象化接口等。

空间数据处理层负责实现对空间数据的存储管理和查询访问。查询语言多是以结构查询语言 (SQL) 为基础,增加相应的函数实现对空间对象和空间关系的查询。查询操作主要由空间选择操作和

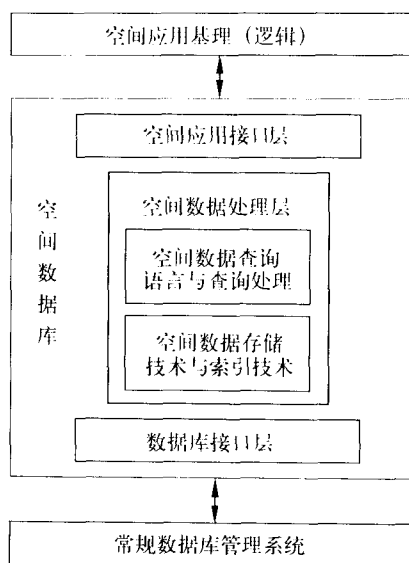


图 1 空间数据库系统结构图

空间连接操作构成。为了提高访问效率,引入了针对空间数据的索引结构。常用的索引结构可分为面向空间点的索引结构(如:网格文件、K 维树、自适应 K 维树等)和面向矩形的索引结构(如:R 树、四叉树和单元树等)。空间选择操作用于查找满足某个空间约束的空间对象的集合。空间连接是查找满足某个空间约束的空间对象对的集合。数据库接口层负责与底层数据库系统之间的融合,处理常规数据的查询,并与空间数据处理层一同实现对空间对象的查询。

参考文献

1. Guting R H, Fernuniverstat H. Spatial Database System vldb'94
2. Shekhar S, Chawla S, etc.. Spatial Databases: Accomplishments and Research Needs. IEEE Transactions on Knowledge and Data Engineering, 1999, 11(1)

(汪卫 施伯乐)

kongzhigan

控制杆 (joystick) 一种用手操纵来直接控制屏幕上光标移动的杆状输入设备。它的外形是一个细杆,装在方形的底座上。杆长约 2.5 ~ 10 cm,杆上有手把,用手抓杆可以前后左右晃动。底座内有由电位器、开关、光学编码盘等构成的传感器,用来检测杆的运动方向、位移和速度。当使用者操纵

此杆偏离其中心位置时,屏幕上的光标也作相应方向和位移的运动。手把上有按钮,用来执行某种操作。

控制杆大量应用于游戏机,这时它是一个相对定位的设备。控制杆也用于工业控制和军事方面,这时有的控制杆带有恢复弹簧,当不受力时会回到中心位置,因此也可以作为绝对定位设备。

控制杆精度低,但操作方便,价廉。

在个人计算机中,控制杆用电缆与主机连接,在主机板上有专用的端口。但随着半导体芯片集成度的提高,已不需要用整个插件做游戏端口了,此端口通常和异步通信电路等做在一块卡上。(林兼)

kuaikeca biancheng zhidu cunchuqi
xinpian

快可擦编程只读存储器芯片 (flash erasable programmable read only memory chip)

一种基于单管存储单元结构的电可修改的半导体只读存储器芯片。简称 Flash EPROM 芯片。Flash EPROM 芯片在功能上与电可擦编程只读存储器芯片 (EEPROM 芯片) 等同,能在电路系统中擦除和写入数据 (编程),但制作时使用与可擦编程只读存储器芯片 (EPROM 芯片) 相同的工艺,采用相同的单管存储单元和电路,因之具有单管的高集成度和可采用塑料封装等低成本的特点。

过去“flash”作为技术专用名词是表示一次可写入数据 (编程) 或可擦除一个完整存储阵列的工作方式,即现在的体写入数据 (编程) 或体擦除方式。虽然后来工业上已设计和制作出可以部分擦除 (块、页擦除) 的存储阵列,但 flash 一词仍继续使用,并且专门指那些基于单管存储单元结构的高集成度电可擦编程只读存储器芯片,而不是常规的 EEP-

ROM 芯片。EEPROM 芯片由于使用了双管或多管存储单元而限制了集成度的提高。

Flash EPROM 芯片大致可分为主要具有程序储存和执行功能的 NOR 体系结构和主要具有数据储存能力的 NAND 和 AND 体系结构。NOR 体系产品的发展主要由手机和个人数字助理 (PDA) 等的应用所带动,已经有 1 Gb 的产品问世。以储存数据为主要应用的 NAND 和 AND 产品是最具规模和发展最快的品种,具有极其广阔的应用前景,包括社会生活中使用的各种存储卡、随身电子盘等。1984 年制作出具有 NAND 体系结构的 256 Kb Flash EPROM 芯片,到 1988 年发展为 512 Kb,1989 年为 1 Mb,1991 年为 4 Mb,1994 年为 32 Mb,1999 年为 128 Mb,2002 年为 1 Gb,2003 年发布的为 0.07 μ m 工艺的 4 Gb 产品。

很多早期的 Flash EPROM 芯片使用改进的 EPROM 芯片单管存储单元,但只能进行体擦除,并且需要 EPROM 芯片一样的外部高写入数据 (编程) 电压。其擦除次数与 EPROM 芯片相当,但比 EEPROM 芯片小几个数量级。这些产品主要以嵌入芯片的形式嵌入到各种过程控制和中央处理器中。新一代 Flash EPROM 芯片可以选择体擦除和多种块擦除方式,存储工艺的发展使芯片的擦除和写入数据 (编程) 次数从早期的约 100 次提高到如今的 1 000 000 次,已经与 EEPROM 芯片相当。Flash EPROM 芯片编程电压有 5 V 和 12 V 两种,采用 5 V 电压的芯片面积大,采用 12 V 的芯片虽然占用的芯片面积较小,却需额外的写入数据 (编程) 电压。后来有低电压的 Flash EPROM 芯片面世,可在 3V 读出和 5V 擦除。

表 1 列出 4 种只读存储器芯片的电气和工艺特性。

表 1 Flash EPROM 芯片等 4 种 EPROM 芯片电气特性和工艺特性对比

特性	常 规 芯 片			新 型 芯 片
	UV EPROM	OTP EPROM	EEPROM	Flash EPROM
封装	带石英窗口的陶瓷封装	塑料封装	塑料封装	塑料封装
擦除时间	20 ~ 30min	不能擦除	1ms	100ms
编程时间	< 1ms	< 1ms	< 1ms	0.2ms
在系统中重写	不能	不能	能	能
可靠性	可筛选	不能筛选	可筛选	可筛选
擦除器	紫外光擦除	不需要擦除	电擦除	电擦除
结构	双多晶硅	双多晶硅	双多晶硅	三多晶硅

Flash EPROM 芯片的规模应用始于 1990 年,由于生产工艺简单、集成度高、功耗低以及能在系统中再写数等特点,正以较快的速度发展。预计会逐渐取代紫外线可擦编程只读存储器芯片(UV EPROM 芯片)和一次可编程可擦编程只读存储器芯片(OTP EPROM 芯片)的市场,并占领一部分传统的只读存储器芯片(ROM 芯片)市场。Flash EPROM 芯片的写入速度比静态随机存取存储器芯片(SRAM 芯片)慢得多,但在低速应用和以读出操作为主的应用场合,完全可以替代电池备份的 SRAM 芯片。

图 1 为 1Mb Flash EPROM 芯片电路框图。其中的地址和控制输入都增加了锁存功能,控制信号与内部的计时器相配合用于控制写入和擦除。该电路的特点是增加了辅助控制电路以调整内部操作的定时关系,使得能有效地减少单元的电力,提供写入数据(编程)和擦除所需波形以及禁止非必需的写入和擦除。芯片的功能选择由控制引脚 \overline{CE} 、 \overline{OE} 和 \overline{WE} 决定,可以选择:读出操作、待用、字节写数、擦除选择、体擦除和页擦除。

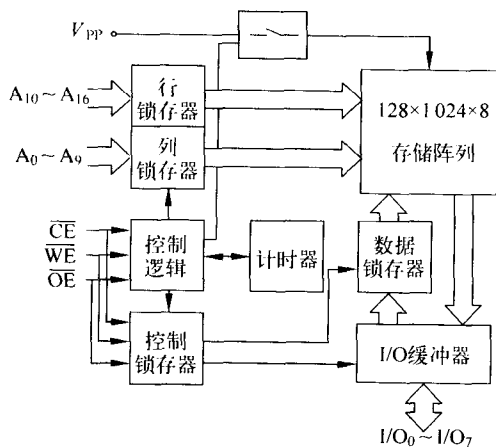


图 1 1Mb CMOS Flash EPROM 芯片电路框图

参考文献

1. Prince B. Semiconductor Memories. 2nd ed. Chichester: John Wiley & Sons, 1991
2. Miyawaki Y. A New Erasing and Row Decoding Scheme for Low Supply Voltage Operation of 16 Mb and 64 Mb Flash EEPROMS. Symposium on VLSI Circuits, June, 1991: 85 (时万春)

kuaisu fuliyie bianhuan

快速傅里叶变换 (fast Fourier transform,

FFT) 计算离散傅里叶变换(DFT)的一种快速算法。

一维 DFT 是把 N 点序列 $x(n)$ ($n = 0, 1, \dots, N-1$) 按线性关系

$$X(k) = \sum_{n=0}^{N-1} x(n) W^{nk}, \quad (k = 0, 1, \dots, N-1) \quad (1)$$

变为 N 点序列 $X(k)$ ($k = 0, 1, \dots, N-1$) 的一种线性变换,其中 $W = e^{-i2\pi/N}$, $i = \sqrt{-1}$,它具有如下性质

$$\frac{1}{N} \sum_{n=0}^{N-1} W^{nt} = \begin{cases} 1, & \text{当 } t \text{ 是 } N \text{ 的倍数时} \\ 0, & \text{其他} \end{cases}$$

利用这个性质,可知逆变换(IDFT)是

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W^{-nk} \quad (n = 0, 1, \dots, N-1) \quad (2)$$

直接计算 N 点 DFT 和 IDFT 各需要 N^2 次乘法和 $N(N-1)$ 次加法,当 N 很大时,运算量非常大。1965 年美国 J. W. Cooley 和 J. W. Tukey 提出的 FFT 算法使运算量降为 $N \log_2 N$, 计算效率提高 $N/\log_2 N$ 倍,引起了各国学者的广泛注意。20 多年来,各种快速算法相继出现,成为数值代数中最活跃的一个领域。

FFT 算法的基本思想是利用基函数 W^{nk} 的周期性和对称性,改变式(1)的计算次序和求和次序,利用递推步骤以减少运算量。例如,当 $N = 2^l$ 时,改变式(1)的求和次序,使偶数项和奇数项分别求和,并对 $X(k)$ 的前、后 $N/2$ 个样本分别计算,就可得到

$$\begin{aligned} X(k) &= G(k) + W^k H(k) \\ X(k + N/2) &= G(k) - W^k H(k) \end{aligned} \quad (k = 0, 1, \dots, N/2 - 1) \quad (3)$$

$$\begin{aligned} \text{其中 } G(k) &= \sum_{n=0}^{N/2-1} x(2n) W^{2nk} \\ H(k) &= \sum_{n=0}^{N/2-1} x(2n+1) W^{2nk} \end{aligned}$$

$$(k = 0, 1, \dots, N/2 - 1) \quad (4)$$

由此可知,只要先计算式(4)中的两个 $N/2$ 点序列 $G(k)$ 和 $H(k)$, 然后由式(3)用 $(N/2 - 1)$ 个乘法和 N 个加法便可计算出 $X(k)$, 而 $G(k)$ 和 $H(k)$ 是 $N/2$ 点 DFT, 可用类似方法计算,如此递推计算下去, $(\log_2 N - 1)$ 步后就可变成二点 DFT。如果用 $M(N)$ 和 $A(N)$ 分别表示 N 点 DFT 所需乘法和加法数,按上述算法有

$$M(N) = \frac{1}{2} N \log_2 N - N + 1 \quad (5)$$

$$A(N) = N \log_2 N \quad (6)$$

其中利用了 $M(2) = 0, A(2) = 2$ 。通常把这种方法叫做时间抽取基-2FFT算法。还有频率抽取基-2FFT算法,这只要将式(1)的 $X(k)$ 按下标 k 为奇数、偶数分别计算,将和式分做前后各 $N/2$ 项相加,就有

$$\begin{aligned} X(2k) &= \sum_{n=0}^{N/2-1} g(n) W^{2nk} \\ X(2k+1) &= \sum_{n=0}^{N/2-1} h(n) W^{2nk} \\ (k &= 0, 1, \dots, N/2-1) \end{aligned} \quad (7)$$

其中 $g(n) = x(n) + x(n + N/2)$

$$\begin{aligned} h(n) &= [x(n) - x(n + N/2)] W^n \\ (n &= 0, 1, \dots, N/2-1) \end{aligned} \quad (8)$$

即先用 $(N/2-1)$ 个乘法和 N 个加法计算式(8)中的 $g(n)$ 和 $h(n)$,再计算式(7)中的两个 $N/2$ 点 DFT,可用相同办法计算,如此递推计算下去, $(\log_2 N - 1)$ 即得结果,运算量也如式(5)和(6)所示。

由基-2FFT算法容易推出任意基- r FFT 算法。设 $N = r^l$ (r 为任意正整数),按时间抽取法,式(1)可化作

$$\begin{aligned} X(k + sN/r) &= \sum_{l=0}^{r-1} G_l(k) W^{ksN/r} \\ (s &= 0, 1, \dots, r-1; k = 0, 1, \dots, N/r-1) \end{aligned}$$

其中

$$G_l(k) = H_l(k) W^{lk}$$

$$H_l(k) = \sum_{n=0}^{N/r-1} x(rn + l) W^{rnk}$$

$$(l = 0, 1, \dots, r-1; k = 0, 1, \dots, N/r-1)$$

也就是把 $N = r^l$ 点 DFT 简化为 N/r 个 r 点 DFT 和 r 个 N/r 点 DFT 来计算,附加运算量仅为 $(r-1)(N/r-1)$ 次乘法,如此递推 $(\log_2 N - 1)$ 步就成为 r 点 DFT,所以基- r FFT 运算量是

$$M(N) = \frac{1}{r} N \log_r N [M(r) + r - 1] - N + 1,$$

$$A(N) = \frac{1}{r} NA(r) \log_r N,$$

其中 $M(r)$ 和 $A(r)$ 是 r 点 DFT 的乘法和加法量。

当 N 不是单一整数的乘幂,而可分解为 $N = N_1 \cdot N_2 \cdot \dots \cdot N_r$ 时,又可类似地建立运算量不超过 $N(N_1 + N_2 + \dots + N_r)$ 的 FFT 算法。例如 $N = 75 = 3 \times 25$,可将 75 点 DFT 简化为 3 个 25 点 DFT 和 25 个 3 点 DFT,而 25 点 DFT 可简化为 5 个 5 点 DFT,因此只要知道 3 点和 5 点 DFT 的快速算法,就可得到 75 点 DFT 的快速算法,这种方法称为混合基 FFT

算法。

上述算法可应用于 IDFT。FFT 算法在数值计算中的一个重要应用是计算循环卷积。此外还可用于快速计算多项式乘积、大整数乘积、某些矩阵的逆和特征值等,FFT 在函数逼近论中和在众多实际技术领域中也已有广泛的应用,在光谱、声谱、地震谱分析、晶体结构分析、滤波、数字信号处理、图像信号处理、物探、雷达、卫星摄像分析、全息图,以及心电图、脑电图、X 光相片强化等方面都有大量应用,很有发展前途。

自从 Cooley-Tukey 提出上述 FFT 算法以来,近十几年又提出了不少新的快速算法,目的是进一步减少运算量以提高计算效率。其中重要的有 Rader-Brenner 算法、Z 变换算法、基-3 新算法、递归割圆分解算法、分裂基算法、余弦变换算法、Winograd 算法以及素因子算法等等。在多维 DFT 的快速算法方面,有以一维 FFT 算法为基础的行列算法和嵌套算法、向量基算法、多项式变换算法等。

参考文献

1. Nussbaumer H J. Fast Fourier Transform and Convolution Algorithms. Berlin, Heidelberg, New York: Springer-Verlag, 1981
2. 蒋增荣,曾泳泓,余品能编著.快速算法.长沙:国防科技大学出版社,1993 (蒋增荣)

kuaisu yitaiwang

快速以太网 (fast Ethernet) 与以太网采用带碰撞检测的载波侦听多址访问-冲突检测 (CSMA/CD) 方法进行介质访问控制方法相同而传输速率为 100 Mb/s 的一种高速局域网。以太网所使用的协议分析及管理工具和在其上运行的高层应用软件均可毫无改变地在快速以太网上运行。快速以太网与以太网的主要区别是在物理层上采用了新的信号编码方式,以便能在光纤和双绞线上以 100 Mb/s 的数据传输速率传送数据。图 1 表示快速以太网与以太网的异同处。

从图 1 中可以看出,在 IEEE 802.3 以太网基准(参考)模型中,与开放系统互连基准(参考)模型中数据链路层相对应的是介质访问控制 (MAC) 和逻辑链路控制 (LLC) 两个子层。在这两个子层及更高层上,快速以太网和以太网都是完全相同的,其区别仅仅在物理层。例如,快速以太网用介质无关接口 (MII) 替代了以太网的连接部件接口 (AUI)。在快速以太网中不使用 AUI 而使用 MII 的主要原因是:

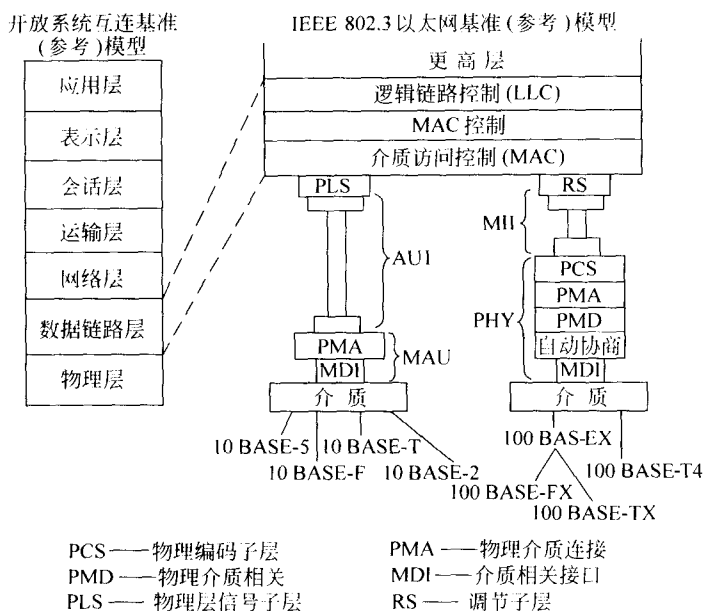


图1 快速以太网与以太网的异同

使用这一层的主要目的是减弱介质访问控制子层对物理层的要求。虽然在 10 Mb/s 的数据传输速率下, AUI 能很好地完成这一功能, 但当数据传输速率提高到 100 Mb/s 时, 采用位串行接口的 AUI 就无能为力了。而 MII 是采用 4 位并行接口, 从而降低了所需频率(此时可从位串的 100 MHz 降到 25 MHz)。此外, MII 还附加了一些其他的新功能, 既能适合在 100 Mb/s 下工作, 亦能在 10 Mb/s 下工作。而 AUI 只能在 10 Mb/s 下工作。

对传统以太网的另一个区别是快速以太网用调节子层(RS)替代了物理层信号子层(PLS)。10 Mb/s 以太网采用物理层信号子层完成曼彻斯特编码与解码。这种编码在高频时, 电磁干扰和射频干扰会变得很严重, 因此在 100 Mb/s 传输速率时就不宜再用曼彻斯特编码, 而只能使用其他的编码方案(参见数据编码)。调节子层(RS)处在 MAC 与 MII 之间, 作为一个“垫片”将 MAC 的位串转换为 MII 的 4 位并行接口。

快速以太网比以太网多了物理编码子层(PCS)和物理介质相关(PMD)子层。PCS 的主要工作是对要发送的数据进行编码和对接收到的数据进行解码。而 PMD 子层主要定义物理层信令, 介质相关接口(MDI)及所支持的传输介质类型。

图1中还列出了快速以太网标准所支持的传输

介质的类型。快速以太网标准称为 IEEE 802.3u, 又称为 100 BASE-T。它又分 100 BASE-X 和 100 BASE-T4 两个选项。100 BASE-X 标准参考了 FDDI 标准中的物理介质规范。100 BASE-X 规定所有传输介质使用单条链路(单条双绞线, 单条光纤)获得单方向的 100 Mb/s 的数据速率。100 BASE-X 规定使用两种物理介质, 即双绞线(100 BASE-TX)和光纤(100 BASE-FX)。为了获得更高的传输效率, 100 BASE-X 标准利用原来为 FDDI(参见光纤分布数据接口)定义的 4B/5B NRZI 编码方案。4B/5B 编码的效率为 80%。为了保证传输过程的同步, 还需要对 4B/5B 编码信号再次用不归零 NRZI 进行编码(参见数据编码)。

为了充分利用建筑物中已安装好的 3 类非屏蔽双绞线(UTP), 快速以太网标准选项中特定义了 100 BASE-T4 标准(参见双绞线电缆)。

在快速以太网中, 全双工操作是一项必备的功能(参见串行传输), 它不仅为了提高带宽(理论上数据速率可达 200 Mb/s), 更重要的目的是要延长网络的最大碰撞域直径。碰撞域直径反映了以太网引入了 CSMA/CD 后对连接介质的长度限制。因为当数据传送速率达到 100 Mb/s 时, 时间槽或者说碰撞域比以太网缩小了 10 倍, 因而碰撞域直径也变成以太网的十分之一。这样的地理覆盖范围限制了快

速以太网的实际应用。因而为快速以太网制定了全双工操作标准,称为 IEEE 802.3X,于 1997 年 3 月正式成为标准。快速以太网以全双工方式操作时,不再使用带碰撞检测的载波侦听多址访问的介质访问控制方法,因而也不受以太网碰撞域影响而造成的作用距离限制,拓展了快速以太网的应用范围。

参考文献

高传善等译校. 局域网与城域网(第 5 版). 北京:电子工业出版社,1998 (方起兴)

kuandai jieru jishu

宽带接入技术 (wideband access technologies) 将主干网连至最终用户,速率高于兆位的接入网技术。当前主干网正向超高速和超大容量的方向发展,高达 160 Gb/s 的波分复用系统已投入应用。另一方面最终用户大量使用的个人计算机处理能力也飞速增长。但连接主干网和最终用户的接入网仍处在窄带水平。因此,接入网的宽带技术成为发展宽带网的关键。下面介绍宽带接入的几种类型:

(1) 不对称数字用户专用线(ADSL) 这是在中继的用户环路上使用有负载电话线提供高速数字接入技术,对少量使用宽带业务的用户是一种经济快速的接入方法(参阅数字用户专用线)。

(2) 高比特率数字用户专用线(HDSL) 这是在中继的用户环路上使用无负载电话线提供高速数字接入技术,可实现在双绞线上高带宽双向传输。

(3) 甚高速数字用户专用线(VHDSL) 这是在 ADSL 基础上发展起来的,可在很短的双绞铜线上传送比 ADSL 更高速的数据。其最大的下行速率为 51~55 Mb/s,传输线长度不超过 300 m(参阅数字用户专用线)。

(4) 混合光纤同轴电缆(HFC) 这是基于现有有线电视(CATV)网基础上的光纤光缆和同轴电缆混合的宽带接入技术。HFC 网是宽带接入技术中最先成熟和进入市场的。HFC 在一个 500 户左右的光节点覆盖区可以提供 60 路模拟广播电视,每户至少两路电话,速率为 10 Mb/s 的数据业务(参阅混合光纤同轴电缆)。

(5) 同步光纤网(SONET) 这是一种高速、宽带主干网技术(参见同步光纤网),在接入网中应用 SONET 的主要优势在于:①对于要求高可靠、高质量业务的大用户,可直接用 SONET 系统以点到点或环形拓扑形式与用户相连,提供理想的网络性能和可靠性。②可以增加传输带宽,改进网络管

理能力,简化维护工作,降低运行维护成本。③同步光纤网的固有灵活性使网络运营者可以更快、更有效地提供用户所需的长期和短期业务需求。对于发展极其迅速的蜂窝通信系统采用 SONET 系统尤其合适,因为它可迅速灵活地提供 2 Mb/s 的透明通道。

考虑到接入网成本的高度敏感性和运行环境的恶劣,适用于接入网的 SONET 设备必须是高度紧凑、低功耗和低成本的新型系统。

(6) 以异步传送模式(ATM)为基础的无源光网络(APON) ATM 化的接入网可为用户提供经济高效的多媒体业务传送平台并有效地利用网络资源。APON 是一种结合 ATM 多业务、多比特率支持能力和无源光网络透明宽带传送能力的比较理想的解决方案,代表了面向 21 世纪的宽带接入技术的最新发展方向。APON 能否大量应用的一个重要因素是价格,APON 的目标价格是宽带无源光网络(PON)的 1.5 倍,但提供的业务范围和业务质量远优于宽带 PON。

(7) 宽带固定无线接入 宽带固定无线接入技术主要有以下三类:①多路多点分配业务(MMDS),带宽为 200 MHz;②卫星直播系统(DBS),带宽为 500 MHz;③本地多点分配业务(LMDS),工作在毫米波段,大致在 28 GHz 附近,可用带宽至少为 1 GHz。

宽带固定无线接入技术代表了宽带接入技术的一种新的不可忽视的发展趋势,它具有敷设开通快、维护简单、用户密度大时成本低的特点,是传统电信业务的有力竞争者。

参考文献

韦乐平. 宽带接入技术的新发展. 电子科技导报,1999(1) (胡道元)

kuandai jieru tixi jiegou

宽带接入体系结构 (broadband access architecture) 主干网连至最终用户的宽带接入网的体系结构。流行的宽带接入体系结构有如下若干种:集成数字环载波(IDLC)、混合光纤同轴电缆(HFC)、光纤到路边(FTTC)、多信道多点分布服务(MMDS)、本地多点分布服务(LMDS)、直接广播卫星(DBS)。

参考文献

1. 胡道元. 智能建筑计算机网络工程. 北京:清华大学出版社,2002

2. Padmanand warrier, Balaji Kumar, XDSL Architecture. McGraw-Hill Inc., 2000 (胡道元)

kuandai jieruwang

宽带接入网 (broadband access network)

将主干网连至最终用户,速率高于兆位的接入网。当前主干网正向超高速和超大容量的方向发展,而最终用户大量使用的 PC 处理能力也迅速增长。但连接主干网和最终用户的接入网仍在窄带水平。因此宽带接入网成为发展宽带网的关键(参见计算机网络)。通常称解决最后一公里的问题。

整个宽带网络可以分成传输网、交换网和接入网 3 大部分,所以宽带网的相关技术也分为 3 类。
①传输技术 宽带网络中要解决的问题是大容量、长距离的可靠传输,采用的物理传输介质是光纤。在传输网体系结构方面,采用光纤同步数字体系(SDH)(参见时分多路复用),SDH 具有标准的网络接口和单元,具有强大的网络管理和维护功能,能灵活支持多种业务。
②交换技术 宽带网络中的交换技术要求能提供高速大容量的交换,能支持各种业务,目前最有前途的交换网络是 ATM 网。
③接入技术 宽带网络对接入技术的要求也是两个方面:网络的宽带化和业务的综合化。接入技术是宽带网络中一项重要技术,它为用户提供一个端到端的宽带连接,并且能通过一条线路、一个接口得到宽带网提供的各种业务。因此,宽带接入网的体系结构(参见宽带接入体系结构)和宽带接入技术受到了极大的重视。

所谓接入网是指主干网到用户终端之间的所有机线设备,如图 1 所示。国际电信联盟在 ITU-T 中规定,接入网是指由业务结点(SNI)和相关用户网络接口之间的一系列传送实体(诸如线路设施和传输)所组成,为传送业务所需传送承载能力的实施系统。它可以经由 Q3 接口进行配置和管理。传送

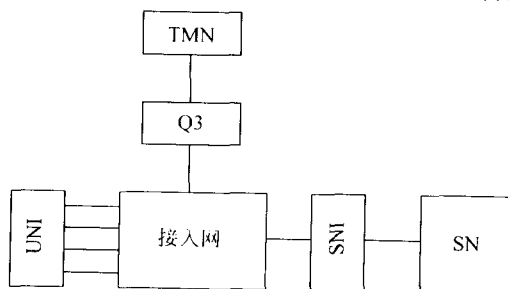


图 1 接入网的接口

实体提供必要的传送承载能力对用户是透明的,不做处理。它可以看作与业务和应用无关的传送网,主要完成交叉连接、复用和传输功能。

接入网所包括的范围由 3 个接口所标志。在主干网侧通过结点接口(SNI)与业务结点(SN)相连;在用户侧经由用户网络接口(UNI)与用户终端相连;而管理功能则通过 Q3 接口与电信管理网(TMN)相连。

接入网的宽带接入技术大致有 4 种技术方案。它们分别是以现有电话网铜线为基础的数字用户专用线(DSL)技术,以电视电缆为基础的电缆调制解调技术,光纤网络及无线卫星接入技术。

参考文献

班洁敏,梁彩隆等. 三级网络技术. 北京:清华大学出版社,2004 (相士俊)

kuangjia biaooshi

框架表示 (frame representation) 由若干个结点和关系构成的网络,用来表示某类情景的结构化的一种数据结构。是语义网络的一般化(参见语义网络表示)。

框架理论是由美国麻省理工学院 M. Minsky 于 1975 年提出的一种知识表示理论,作为理解视觉、自然语言对话以及其他复杂行为的一种基础。当我们对事物进行观察时,外界事物、事件等在头脑中形成特定的概念,并在记忆中形成有关某物的整体结构。这一结构就称之为框架。框架提供了一个对象或一类对象的结构化表示,可以描述典型状况、事实、对象和概念及它们之间的层次关系。这种表示既是层次化的,又是模块化的,是一种理想的结构化知识表示方法。用它来表示有关事物的知识时,不仅可以表示出事物各方面的属性,而且可以表示出事物之间的类属关系,事物的特征和变异等等,在识别、分析、预测事物及其行为方面有很大的用处。当遇到新情况或换一角度看问题时,往往会把老的框架与新的情况作比较,把新的数据填入该框架中便形成一个特定概念,并且人们还常常根据典型的框架去预测所属范围内的一个未来情况。

在框架结构中填入有关新情况的数据时,可以根据以往的经验获得的概念对这些数据进行分析和解释。另外,这种框架结构也提供了在一具体的上下文中对特定实体进行预见驱动的处理方式,在此上下文环境下根据这种结构可以寻找那些预见的信息。框架结构中存放这些可预见信息的位置称为

槽。一个框架由若干个槽组成,每个槽有它自己的名字。槽内的值描述框架所表示的实体的各个组成部分的各种属性,每个槽的值又由一个或多个侧面组成。各侧面从各个方面来描述槽的特性,每个侧面又有一个或多个侧面值,作为对槽的进一步说明。每个侧面值可以是一个值或者一个概念的描述。

框架的一般形式可以表示如下:

框架名:

槽 - 1:	侧面 - 11	侧面值 - 11
	
	侧面 - 1m	侧面值 - 1m
槽 - n:	侧面 - n1	侧面值 - n1
	
	侧面 - nm	侧面值 - nm

槽或侧面的取值可以是二值逻辑的真或假;可以是实数值;可以是文字以及其他形式的定义域,也可以是一组子程序,称之为框架的程序附件。

在框架系统中框架匹配的含义即是把一个实体的一组值与一个框架的期望值进行比较。由框架所

构成的知识库,当利用它进行推理、形成概念和作出决策、判断时,其过程往往是根据已知的信息,通过与知识库中预先存储的框架进行匹配,找出一个或几个与该信息所提供情况最适合的预选框架,形成初步假设,即由输入信息激活相应的框架,然后在该假设框架引导下,收集进一步的信息。按某种评价原则,对预选的框架进行评价,以决定最后接受或者放弃预选的框架,即在框架引导下的推理。框架系统使用多种类型的推理:如默认推理、存在性推理、公共属性推理、异常情况确认推理和类比推理等。

综上所述,框架表示法为概念、结构和功能模型等陈述性知识的描述提供了一种结构化的典型方法,但对过程性知识的表达能力还比较差,将框架表示与产生式表示结合在一起能较好地解决这一问题。

参考文献

Minsky M. A Framework for Representing Knowledge. The Psychology of Computer Vision, New York: McGraw-Hill, 1985
(陈世福)

L

lei

类 (class) 一组具有共同特性的相似对象的抽象描述。类是面向对象语言的基本成分,它可进一步理解为:①面向对象程序的惟一构造单位;②抽象数据类型的具体实现;③对象的生成模板。

类与对象 面向对象程序由一组相关的类构成,故类是静态的;程序的执行体现为一组相互通信的对象的活动,故对象是动态的。类是一组具有共同特性的相似对象的抽象描述,对象是其具体实例。

作用 在面向对象语言中,类的作用有二:一是作为对象的抽象描述机制,类刻画相似对象的共同属性和行为;二是作为程序的基本构造单位,类支持模块化设计,其分类关系是模块划分的规范标准。

种类 ①按定义方式分为系统类和用户类:前者由系统内部定义;后者由用户自行定义;②按继承关系分为基类和衍类;后者是通过继承前者的属性和操作而定义。衍类也称子类,基类也称父类。需要由其衍类进一步定义类称为延迟类或虚拟类,它无对象实例。

类间关系 类间的主要关系是构成类间层次结构的继承关系。

参考文献

1. 徐家福等. 对象式程序设计语言. 南京: 南京大学出版社, 1992

2. Special Issue on Object-Oriented Design. CACM, 1990, 33(9) (张家重 王志坚)

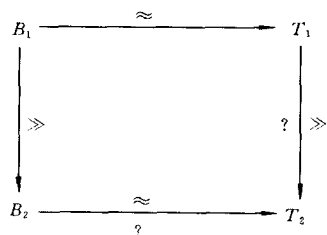
leibi tuili

类比推理 (analogical reasoning) 由于认识到当前新情况与已知熟悉情况在某些方面相似,从而推出两者在其他相关方面也相似的推理方式。

这里情况包括人们曾经经历的事件、求解的问题和解释的现象等。它来自对人类记忆的研究,成长于对基于规则系统的改进。当前人工智能研究已表明基于规则系统的如下缺点:难于进行知识获取,凡事都从基本原理开始推理是低效的,而且离开了规则库就寸步难行。认知研究表明,在遇到新情

况时,人们时常回忆相似情况的处理经验,经适当修改后作为新情况的处理策略。显然,类比推理能较好地克服基于规则系统的上述缺点。

在类比推理中,通常把已知熟悉情况称为基或源,把新情况称为靶。类比推理的推理模式如下(?表示推出部分):



即已知基中知识 B_1 和 B_2 , 靶中知识 T_1 , B_1 与 B_2 相关(或 B_2 依赖于 B_1), T_1 与 B_1 相似;在靶中推出类比结论 T_2 , 这里 T_2 与 T_1 相关、与 B_2 相似。显然,如果将其中的“ \approx ”改为“ $=$ ”(相同),“ \gg ”改为“ \supset ”(真值蕴含),则上述类比推理模式变为演绎推理模式 $T_1, T_1 \supset B_2 \vdash T_2$ (其中 $T_2 = B_2$), 这种变化说明:

(1) 类比推理是似然推理。因为“ \approx ”比“ $=$ ”弱,“ \gg ”比“ \Rightarrow ”弱。

(2) 为提高类比结论的可信度,希望:

B_1 与 T_1 间的相似度高,这使 $B_1 \approx T_1$ 接近于 $B_1 = T_1$;

B_1 包括的本质特征多,这使 $B_1 \gg B_2$ 更好地反映了情况组成元素间的本质联系,即接近于 $B_1 \Rightarrow B_2$ 。

虽然研究者们对类比推理过程进行了不同的划分,但是通常都经历了下列步骤:

(1) 分析靶的描述,从中抽取用于检索相似情况的特征,也把这些特征称作索引;

(2) 根据索引在情况库中检索与靶最相似的情况,这些情况就是基;

(3) 建立基和靶的组成元素间的对应关系,通常是使功能或作用相似的元素对应起来;

(4) 根据已建立的元素对应关系,将其中未被对应的知识转换到靶,从而得到类比结论。

基于不同的出发点,可对类比推理进行不同的

分类。例如,根据情况间的相似程度可分为领域内类比和领域间类比;根据处理任务的类型又可分为类比问题求解和类比解释。

关于类比推理的特点和基本问题可参见**类比学习**。
(李波)

leibi xuexi

类比学习 (learning by analogy) 以**类比推理**为基础,通过识别两个情况的相似性,来从一种情况中的知识去分析或理解另一种情况的**机器学习**方法。

在这两个情况中,一个是已经理解的熟悉情况,称为基或源,另一个是待理解的新情况,称为靶。这里“情况”可以是概念、任务、事件等。日常生活和科学发现中有大量类比学习的例子。例如,根据茅草上又长又密的锋利茅刺会划破手,鲁班想到了有齿的铁条能锯树,从而发明了锯子。根据氢原子与太阳系之间的相似性,卢瑟福提出了氢原子模型。此外,在客观实践和科学实验中,由于种种限制有时会使得人们无法直接考查某些事物,这时类比学习可以提供一种很好的间接实践手段。当前,类比学习已经成为人工智能,特别是机器学习研究的重要组成部分。

类比学习的研究始于60年代后期,这一期间的工作主要集中在数学领域的类比学习。具有代表性的工作有平面几何智力测验系统 ANALOGY,辅助定理证明系统 ZORBA,以及 P. H. Winston 的先例学习系统 Fox。自80年代以来,对类比学习进行了较为全面和深入的研究。在类比认知机制方面进行了大量实验,主要包括类比在不同年龄的人的学习中的作用和相似性对人类类比的影响。针对人类类比的某些特征已提出了一些类比学习原理,例如 D. Gentner 等人的结构映射理论等。在类比学习计算模型研究方面集中了较多工作, J. R. Carbonell 在提出了基于手段-目的分析法的计算模型之后又提出了推导类比。 M. H. Burstein 利用因果模式构造了模拟学生学习赋值语句的系统 CARL。此外,还对类比学习的匹配和转换阶段进行了一些形式化工作。

类比学习的基本过程 接受一个新情况(靶),检索相似情况,将相似情况的处理转换到靶,存储靶及其处理。虽然不同的类比学习系统强调这一过程的不同阶段,但大多数系统都不同程度地讨论了下列步骤:

(1) **抽取特征** 当输入靶后,需要对其分析,以抽取一组用于寻找相似情况的特征,在类比学习中把这些特征称作索引。

(2) **检索相似情况** 应用第一步中计算出的索引从记忆中检索靶的相似情况,即基。基与靶的相似程度越高,越有助于靶的处理。

(3) **建立对应关系** 建立基和靶的组成元素间的对应关系,通常要求相对应的元素在基和靶的共享因果关系网中有相似作用。

(4) **转换知识** 以第3步建立的对应关系为依据,选择基中未被对应的知识并转换到靶,从而生成类比结论。通常要求被转换的知识与已对应知识间有因果联系,而且类比结论既不能与靶中已有知识矛盾,也应当对靶有用。

(5) **验证类比结论** 由于类比推理是似然推理,因此需要验证类比结论的正确性。

(6) **修改记忆** 从两个方面修改记忆。首先,推广基和靶这对相似情况,得到适合同类情况的一般知识。这是因为许多认知实验都表明经过类比推理后,人们提高了处理类似情况的能力,这说明在类比比较两个相似情况时人们自然地总结出了适合同类情况的一般知识。其次,在记忆中存储靶及其处理,即对它建立索引,以便将来用于其他相似情况的处理。

在上述类比学习过程中存在两种学习。第一种发生在第4步,即在靶中加入类比结论,从而学习到关于靶的新知识,这是类比学习的主要目的。另一种发生在第6步,即得到了适合同类情况的一般知识并记住了靶及其处理,这是类比过程的附带产物。

类比学习的特点 与其他学习方式相比,类比学习有如下主要特点:

(1) **提供了一种自然的学习机制** 类比学习来自对人类记忆的研究,认知实验已提供了人们使用类比学习的许多证据,研究表明类比是儿童和新手经常使用的学习方式。在程序设计和数学课程中,学生常常借助教科书上或自己熟悉的例子来编写新程序和求解新问题。

(2) **更容易进行知识获取** 通常专家容易记住他们曾经求解过的问题,而难于用恰当的规则描述他们求解问题的技术。事实上,许多领域已有大量的现存实例。另外,情况间的交互比规则间的联系少得多,所以更容易对知识库排错。

(3) **有助于提高系统性能** 通过修改以前问题

的解,类比学习系统就可以快速给出新问题的解,避免了从零开始推理的低效率;通过记住过去的错误,就可以避免犯同样的错误。在领域知识不足时,类比学习系统还可以利用相似情况处理新情况。

类比学习的基本问题 由于情况是类比学习的操作对象,相似性是类比学习的基础,这就使得情况的存储与检索、匹配与应用以及相似性的判断准则成为类比学习的基本问题。下面结合当前研究讨论这些问题。

(1) 索引与记忆组织 按适当的特征标记情况就称对情况建立索引。特征选择的合适程度将直接影响类比学习的质量。一般来说,特征的选择有两种策略,一是使用情况的具体特征,如组成情况的对象和描述情况间关系的谓词。这种方案的优点是简单;缺点是难于区别表面相似情况,因而找到的相似情况集中有无关情况。二是使用抽象特征(如目标、约束)作为索引。这种方案的优点是容易找到真正的相似情况;缺点是难于确定索引的抽象层次并带来了额外开销。目前已经普遍认识到没有一种索引方案适合所有系统,因此只能根据任务的特点选择上述一种策略或两种策略并用。由程序设计者确定索引毕竟不是人们所希望的,现在已有一些工作涉及自动索引选择,例如使用基于解释的方法来确定每个情况的相关特征,并把这些特征作为索引。

随着情况的增多,使得情况的组织和管理成为影响检索相似情况的另一个关键因素。传统的情况组织方法是利用区别网,这种方法适合于对某个领域内的情况进行组织。近来在记忆组织方面的一些研究工作支持并行检索方法。

与记忆组织紧密联系的问题是对情况进行整体存储还是分段存储。整体存储方式把情况存储在同一地方,其优点是容易得到情况的完整表示;缺点是难于实现多类学习。分段存储将情况分成多个部分,将各部分存储在记忆的不同地方。这种方式容易实现多类学习,潜在的发现能力强;其代价是在使用情况前需要对其“复原”。

(2) 检索相似情况 由于类比学习是通过从记忆中检索相似情况来实现的,因此怎样快速有效地从记忆中检索靶的相似情况成为其基本问题之一,而且记忆的情况越多,这一问题变得越突出。

相似情况的检索方法与索引的选择和记忆组织方式紧密相关。在区别网组织的记忆中,通常以情况间的推广—具体化层次结构为基础,使用概念求精

技术。即从网顶开始,只有当靶中特征与当前节点匹配时才继续向下搜索。当靶中用作索引的特征都与当前节点匹配时,则当前节点之下的情况集就是检索到的所有相似情况。只有当能在情况间建立正确的区别网时才能使用这种方式,其优点是能很快排除无关情况。

当以低级特征为索引,并把情况简单地存储在记忆中时,通常采用的检索方法分为两步。第一步从靶的低级特征出发,检索一组具有相似低级特征的可能相似情况。第二步比较靶与各个可能相似情况,考查它们的突出(或重要)结构特征是否相似,从而找出真正的相似情况。

近年来出现的并行检索算法是一次检查靶与所有(或多个)已知情况间的相似性,并返回相似度最高的情况。

(3) 类比匹配 建立基和靶的组成元素间的对应关系。通常基和靶间并不精确匹配,所以必须进行部分匹配,这就使得类比匹配问题具有指数复杂性。目前的匹配技术主要有面向基本特征的匹配技术和面向结构特征的匹配技术。在面向基本特征的匹配技术中,首先建立相似基本特征间的对应关系,然后看这种对应关系是否支持结构特征间的对应关系。对应地,面向结构特征的匹配技术首先建立相似结构特征间的对应关系,并进一步得到基本特征间的对应关系。例如,根据两个命题间的对应可以得到其谓词和相同位置参量间的对应。由于在情况的相似比较中,结构特征间的相似比基本特征间的相似更重要,所以后一种匹配技术比前一种更合理。因此,早期的类比学习系统常用面向基本特征的匹配,而近年来的系统则多用面向结构特征的匹配。

为了降低匹配过程的复杂性,一些类比系统利用领域的具体特点,对部分匹配空间采用启发式搜索方法。

(4) 相似性判断 相似性是类比学习的基础,因此怎样判断一个情况与另一个情况的相似性是类比学习的基本问题。从含义、形式和用途三个方面可将类中的相似性分为语义相似、结构相似和目标相似,其中语义相似又包括对象相似和谓词相似。对象相似是指两个对象具有相似的属性,例如“猫”和“狗”都是动物,因此它们是相似的。若两个谓词是同义词或近义词,则称它们是相似谓词。结构相似要求情况组成元素间具有相似的连接关系(或因果联系)。目标相似是指两个情况具有相似的目标。

情况间的相似性是上述四种相似性的加权和。由于一个特征的作用(或重要性)常常随上下文变化,而对象的具体作用和谓词的具体含义隐含在它们参与描述的结构中。因此,通常结构相似和目标相似的权值高于谓词相似和对象相似。

(5) 知识改编 类比学习过程的第5步需要将基中的知识转换到靶。由于基和靶通常并不完全一样,所以需要改编基中的知识才能使之适合于靶。已提出的改编策略有:

① 替换方法 用靶中元素替换被转换知识中的基元素,使之更好地满足靶。简单的作法是将基元素替换成对应的靶元素。但是当某个基元素没有对应的靶元素时,需要先在靶中创建其对应的相似元素。

② 变换方法 根据靶的具体需要,变换基中的知识。常用的实现方式有插入、删除和重排序等。

目前已构作了许多类比学习系统,根据任务类型可将它们分为类比问题求解系统和类比解释系统。前者用于建立新问题的解,常用于规划、诊断和设计任务中。后者用于论证某个结论的合理性或给出引起某种现象的原因,已用于法律辩护、分类和理解任务中。

参考文献

1. Shapiro S C. Encyclopedia of Artificial Intelligence (2nd Edition). New York: John Wiley & Sons, 1992
2. Vosniadou S, Ortony A. Similarity and Analogical Reasoning. New York: Cambridge University Press, 1989
3. Hammond K J. Proceedings: Case-Based Reasoning Workshop (DARPA), III. Calif: Morgan-Kaufmann, 1989

(赵沁平 李波)

leixing dingyi

类型定义 (type definition) 程序设计语言中的类型扩充设施。程序人员使用这种设施可以自行定义所需的类型。在有些语言中类型定义又称**类型说明**。类型定义的作用有二,一是起缩写作用,二是定义新类型。

类型定义的一般形式是:

type T = 类型

这里的类型为已知类型或其中又包括所定义的类型T,前者起缩写作用,后者定义新类型。

例1:

type person = record

name: packed array [1..20] of

char;

age: integer;

height: real

end

例2:

type TP = record

f1: char;

f2: TP

end

(徐家福)

leixing lun

类型论 (type theory) 为避免集合论悖论而建立的,主要研究集合的分层、分类方法(包括公理化方法)的数学理论。

近20年来,它在计算机科学中得到广泛应用。20世纪初 B. Russell 提出了类型论作为公理集合论乃至数学的基础,30年后 A. Church 用 λ 演算和类型的概念定义了高阶逻辑的形式系统。但在经典数学中,类型理论始终未受到广泛关注,直到60年代末才有大的改观。原因有三:第一,20世纪初可构造性数学虽成为数学的一独立分支,但由于 W. Brouwer 开创的直觉主义并没有得到很多支持,一般人的信念是数学的主要部分不可能构造地研究,改变这一信念的是 E. Bishop 的工作,构造性地重建了经典分析的核心部分,表明了构造数学是可以与经典数学相媲美的。60年代 E. Bishop 及其追随者在构造数学领域中作出的重要成果使直觉主义得到发扬光大。直觉主义逻辑的证明论启用了将类型作为基本概念的方法。基本思想是将一个可构造命题看成类型,即类型的元素为该命题的一个证明。这便是“命题当作类型”观点(柯里-霍华德观点,或称解释)。J. Y. Girard 首先使用该原则给出二阶直觉主义逻辑的类型表示,并证明该逻辑是典范化的。P. Martin-Löf 也提出了他的类型理论,为可构造性数学提供了基础。第二,TOPOS 理论的出现,建立了 TOPOS 的内逻辑与直觉主义逻辑两者间的联系,在更广泛的范围内揭示了范畴结构的逻辑特征,对类型理论的认识更深入了。第三,也是最重要的是计算机科学的推动作用,高级程序语言不断涌现,相应的类型系统也得到研究。另一方面,在计算机上实现了用类型论求解数学问题的系统。

以“类型理论”名称出现者,颇为广泛,有分枝

类型论、简单类型论(罗素)、T 理论(K. Gödel)、F 理论和 $F\omega$ 理论(J. Y. Girard)等,其中与计算机科学有关的有以下几种:

(1) 逻辑类型理论,它包含内逻辑。该理论目的是为构造数学提供类型论基础。通常所说的构造演算即是逻辑类型系统。本质上是高阶直觉逻辑的类型表示。逻辑类型系统的著名的子系统有:多态 λ 演算,马丁洛夫类型理论和逻辑框架。

(2) 程序类型理论,研究程序语言的类型系统。新型程序语言、程序设计方法学、语义理论的迅速发展迫切需要类型系统的研究。使用“类型”这一概念大致有两个含义:一指论域的多种组合形式,二指语法范畴的分层结构。该理论涉及的类型研究包括下述基本内容:(表达式 a 有类型 A ,记为 $a:A$,读作 a 为 A 的一个居元。反过来也说 A 有居元 a 。)

不动点操作:设 A 为类型,不动点算子 Y : $(A \rightarrow A) \rightarrow A$,使得对于每个 $f: A \rightarrow A$,有 $Yf: A$ 且 $Y(f) = f(Y(f))$ 。不动点算子是引入不终止计算的基本工具。

归纳类型:类型为 A 的变量 x 是正定的是指 x 只出现在偶数个(函数类型)符号 \rightarrow 的左面,例如 x 在 $x \rightarrow y \rightarrow z$ 中即为正定的。归纳类型产生规则: A 为类型且 x 在 A 中是正定的则 $\mu x. A$ 为类型。许多数据结构如自然数、表、树等均是归纳定义的。

递归类型:归纳类型产生规则中删去正定条件即为递归类型。它与归纳类型差别在于前者是非良序的,后者是良序的。分别加入到二阶 λ 演算中,加入后者时将是强典范化的,递归类型则不然。

多态类型:一程序处理不同类型的输入,有两种多态类型:特定型和参数型。特定型对不同类型的输入反应是任意的无规则的;参数型对不同类型的输入的反应是统一的。例如,项 t 有参数型 $\forall x: \text{type}. A[X]$, t 看成函数,对于类型为 T 的输入给出结果为项 $t(T)$,其类型为 $A[T/X]$ 。

记录类型:记录是程序设计语言的中心概念。对象是记录,属性是记录命名值。记录类型是一组命名类型。例如,记录 $\langle x = \text{True}, y = 2 \rangle$ 有类型 $\langle x: \text{Bool}, y: \text{Int} \rangle$ 。

子类型:直觉上说,类型 A 是类型 B 的子类型当且仅当 A 的每个居元是 B 的居元,记为 $A < B$ 。这个子类型关系满足自反、传递性质。关于函数类型操作有

$$\frac{A < A', B < B'}{A \rightarrow B < A' \rightarrow B'}$$

意即类型 $A \rightarrow B$ 的程序必为类型 $A' \rightarrow B'$ 。继承性在类型论上较为贴切的描述是使用子类型关系。

交类型:对任意两个类型 A, B 可定义交类型 $A \wedge B$ 。无论从证明论、模型论角度看,交类型都是较艰涩的概念。例如设项 f 为类型 $A \wedge (A \rightarrow A)$,则 $f(f)$ 便有定义。因为交类型 $A \wedge (A \rightarrow A)$ 既是 A 的子类型又是 $A \rightarrow A$ 的子类型,故有 $f: A$ 及 $f: A \rightarrow A$,这种自身作用在自身上的项,在一般类型理论中不能合适地定义其类型。

高阶类型:以其他类型作为其居元的类型。例如, $U_0: U_1$ 可解释为:如果 U_0 代表程序的类型,则 U_1 就是程序的规约的类型。

至今已有不少类型系统问世,它们的证明和模型的研究都很有意义。现行的程序设计者总是在追求新的程序设计思想,因此类型的研究是一个具有活力的方向。

参考文献

1. Howard W. The formulae-as-types notion of construction. In: Seldin J P and Hindley J R editor. To Curry H B: Essays on Combinatory Logic, Lambda Calculus and Formalism. New York: Academic Press, 1980
2. Martin-Löf P. A Theory of Types. Report 71 ~ 73, Dept. of Mathematics, University of Stockholm, Feb. 1971, Revised Oct. 1971 (陆汝占)

lisan shijian xitong fangzhen

离散事件系统仿真 (discrete event system simulation) 对离散事件系统建立数学模型,并在计算机上对该模型进行试验的仿真技术。离散事件系统是指由于随机事件的驱动使得系统的状态只是一些离散的时间点上发生变化的系统。

离散事件系统的状态量由随机事件的驱动而发生变化,在两个相邻事件发生之间,系统状态量是保持不变的,即是离散变化的,如订票系统、加工制造系统、交通控制系统等。由于离散事件系统固有的随机性,对这类系统的研究分析往往比较困难。经典的概率及数理统计理论、随机过程理论虽然能对一些简单系统提供解析解,但对大量实际的系统,仍需运用仿真技术来提供较为满意的结果。离散事件系统仿真技术是针对各种离散事件系统所共有的一些特性而产生、发展并得到广泛应用的。

离散事件系统仿真经常采用的几个基本概念

(1) 实体 用于描述系统中的对象。离散事件

系统中的实体可分为两大类,一类为“临时实体”,它只在系统中存在一段时间。一般这类实体由系统外部进入系统,经过系统的处理(服务)后通过系统,最终离开系统,如银行系统中的“顾客”,制造系统中的“零件”均属于临时实体。另一类为“永久实体”,它是永久驻留在系统中的实体,如银行系统中的“服务员”,制造系统中的“设备”,只要系统处于活动状态,这些实体就存在,它是系统处于活动的必要条件。临时实体按一定规律不断地到达(产生),在永久实体作用下通过系统,整个系统呈现出动态过程,因而离散事件系统又称为离散事件动态系统(DEDS)。

(2) 事件 是引起系统状态发生变化的行为。离散事件系统是由事件驱动的。例如,“零件到达”在制造系统中可定义为一类事件,因为由于“零件到达”,系统的状态——某台设备的状态可能从闲变到忙(如果原来状态为闲),或是设备缓冲区状态——排队长度发生变化(队列中零件数加1)。在一个系统中,一般有许多类事件,每一类事件的发生时间往往带有随机性;某一类事件的发生可能引起其他类事件发生,或者是另一类事件发生的条件等。为了实现对系统中的事件管理,仿真模型中必须建立“事件表”,表中记录每一项已发生或即将发生的事件,包括其类型、发生时间及其他有关的属性,以便计算机能仿真实系统中的并行活动。

(3) 仿真钟 用于模拟实际系统的时间属性。系统的动态特性表现为系统状态随时间变化而发生变化,离散事件系统仿真就是要使模型在系统状态发生变化的时间点上仿真出实际系统的动态行为。模型中的时间变量就是仿真钟。仿真过程中,仿真钟的取值称为仿真钟的推进,两次连续取值的间隔称为仿真步长。离散事件系统仿真中的仿真钟推进方法大多采用“下一最早发生事件的发生时间”的方法,也称为事件调度法(参见离散事件系统仿真建模方法学)。由于事件发生时间的随机性,因而仿真钟推进步长也是随机长度,而且,由于相邻两事件之间系统状态不会发生任何变化,因而仿真钟跨过了这些“不活动”周期,从而呈现出跳跃性,其推进速度具有随机性,这是离散事件系统仿真与连续系统仿真的重要区别之一。

(4) 统计计数器 用于仿真模型执行过程中搜集并统计与系统性能有关的数据。离散事件系统仿真的主要目的不仅是要得知系统的状态是如何变化的,更重要的是要得到系统在统计意义下的性能。

因为随机模型的每一次运行只不过是随机过程的一次抽样,只有统计意义下的模型运行结果才有较好的参考价值。

(5) 随机变量模型 用于描述离散事件系统中随机事件的概率分布形式(规律)。由于离散事件系统中随机因素的存在,在系统建模过程中需要采用服从各种分布规律的随机变量来描述系统中存在的随机事件或随机因素。这些随机事件或随机因素具体表现为随机变量的概率分布规律。

常见的随机变量模型有正态分布、指数分布、均匀分布、伽玛分布等。各种已知的概率分布都有其各不相同的特性,因此,选择适宜的概率分布形式,建立合理的随机变量模型是系统建模的一个重要方面。

(6) 伪随机数 是指在计算机上利用数学算法生成的在统计意义上服从 $[0,1]$ 区间上均匀分布的随机变量。

(7) 伪随机变量 是指在伪随机数的基础上生成的具有某种概率分布规律的随机变量。

离散事件系统仿真的一般步骤 离散事件系统仿真的一般步骤可用图1来表示。下面就各步骤的一些特殊问题加以说明。

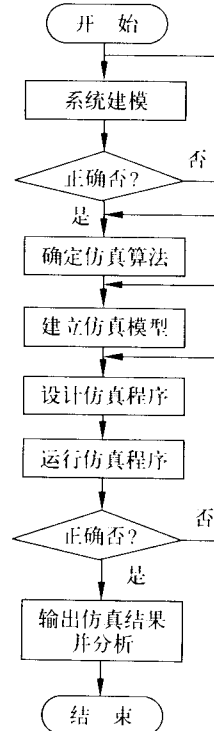


图1 离散事件系统仿真的一般步骤

(1) 系统建模 一般用流程图的形式加以描述。重点描述临时实体的产生规律, 历经系统的过程, 永久实体对临时实体的作用规则、条件及结果。随机变量模型的确定在离散事件系统建模中占有十分重要的地位, 要搜集足够多的原始数据以便较好地确定其分布的类型及参数。对于系统模型应进行有效性检验, 以确定所建系统模型能有效地代表所研究的真实系统。

(2) 确定仿真算法 主要包括两方面的内容, 即从所要求随机变量模型中产生相应的随机变量, 以及选择建模策略(参见离散事件系统仿真建模方法学)。

(3) 建立仿真模型 根据已确定的仿真算法, 建立被仿真系统的计算机模型。仿真模型是系统状态转移的动态描述, 因此首先要定义系统的状态变量, 这要根据系统的内部结构及仿真研究的目的来确定。即使是同一系统, 仿真研究的目的不同, 系统状态定义也可能不尽相同。在离散事件系统中, 状态的变化是由事件引起的, 因此, 要在定义系统状态的基础上定义系统事件及其有关属性。系统事件的定义与建模策略有关, 如采用事件调度法建模, 要定义出系统中的全部事件及相应的事件处理流程, 包括事件类型、事件发生时间、同时发生事件的处理规则等。如采用活动扫描法及进程交互法, 则还要定义相应的活动或进程, 以便按活动或进程的观点来建立仿真模型(参见离散事件系统仿真建模方法学)。

(4) 设计和运行仿真程序 仿真模型最终是通过仿真程序来实现的。可根据所拥有的仿真工具的情况选择某种高级语言(如 FORTRAN 语言, C 语言等)或离散事件系统仿真语言来实现所建立的仿真模型(参见仿真语言)。在仿真试验前, 首先要校验仿真程序的正确性, 然后校验仿真模型的正确性——经过验证是正确的仿真模型才能进行仿真试验。仿真试验条件及运行次数与输出分析的要求有密切关系, 这是与连续系统仿真的重要区别。

(5) 分析仿真结果 对具有随机性因素的离散事件系统模型的一次运行仅仅是随机过程的一次抽样。仿真结果的置信度应予以特别的注意(参见离散事件系统仿真输出分析)。

参考文献

肖田元, 张燕云, 陈加栋. 系统仿真导论. 北京: 清华大学出版社, 2000

(徐文胜)

lisan shijian xitong fangzhen jianmo fangfaxue

离散事件系统仿真建模方法学 (modeling methodology of discrete event system simulation)

建立离散事件系统仿真模型方法的总称。离散事件系统仿真的核心问题是建立描述系统行为的仿真模型。与连续系统的区别在于, 离散事件系统的模型难以采用某种规范的形式, 而一般采用流程图或网络图的形式才能准确地定义实体在系统中的活动。在一个较为复杂的离散事件系统中, 一般都存在诸多的实体, 这些实体之间相互联系, 相互影响, 然而其活动的发生都统一在同一时间基上, 采用何种方法推进仿真钟, 建立起各类实体之间逻辑联系, 这是离散事件系统仿真建模方法学的重要内容, 有时称为仿真算法或仿真策略。

目前, 比较成熟的有 4 种仿真建模方法, 即事件调度法、活动扫描法、进程交互法和三阶段法。在建立仿真模型时, 一般并非只拘泥于采用某一种策略, 同一个仿真模型有时可同时采用几种策略。下面对上述 4 种方法加以说明。

(1) 事件调度法 事件调度法用“事件”的观点来抽象真实系统(参见离散事件系统仿真), 即通过定义事件及每一事件发生对系统状态的影响, 并按事件发生时间顺序来确定每类事件发生时系统中的各实体之间的逻辑关系及其状态, 这就是事件调度法的基本思想。

采用事件调度法建立仿真模型时, 所有事件均按时间先后存放在事件表中; 同时, 模型中要设计一个时间控制部件, 该部件的作用是实现仿真钟的管理与控制。任何一个事件处理模块在执行完后都必须返回到时间控制部件。这样, 事件的选择与处理不断地进行, 仿真钟按事件时间往前推进, 直到仿真终止的条件满足为止。

这种方法的特点是仿真钟的推进仅依据事件发生的时间, 因而, 在建模时有两个基本问题需要特别注意。第一是所谓“同时事件”。因为在任何时刻, 计算机只能执行某一个事件的程序, 所以具有相同发生时间的事件, 模型中必须事先规定其处理顺序, 称为规定“解结规则”。这一般是通过定义事件的优先级来解决的。第二是所谓“条件事件”。在某些系统中, 事件的发生不仅具有时间属性, 还往往带有条件属性。从本质上讲, 事件调度法是一种“预定事件发生时间”的策略, 如果按预定时间某一事件应该发生, 但发生该事件的条件(如果有的话)不

满足,则必须推迟或取消该事件的发生。所有上述两方面的问题,都应在相应的模块中特别加以处理,以免产生模型的死锁。

(2) 活动扫描法 活动扫描法用“活动”的观点来描述实际系统,即通过定义活动及每一活动发生对系统状态的影响来建立系统模型。所谓“活动”就是有关联的两个事件之间的过程。这种建模策略特别适用于对活动持续时间有较强不确定性的系统进行仿真。由于是采用“活动”的观点建模,活动扫描法要求定义系统中所有“活动”及相应处理“活动”的子例程,包括定义活动发生的条件,而活动发生的时间也作为条件之一,只不过它是具有最高优先级的条件。同时,这种建模策略包括了两个控制仿真流程的基本部件,即活动扫描部件及条件处理部件。活动扫描部件的任务是在仿真的每一步对系统中定义的全部活动按优先级从高到低逐个扫描;而条件处理部件则用于对被扫描活动的有关条件进行测试。

若用 $TIME$ 表示系统仿真钟的值, t_{α} 表示某一活动中的实体 α 状态发生变化的时间,活动扫描法的机理可表述为:

首先进行初始化操作,然后活动扫描部件执行活动扫描。当:

$t_{\alpha} > TIME$, 表示该活动在将来某一时刻可能发生,进入下一活动扫描;

$t_{\alpha} = TIME$, 表示该活动如果条件满足则应立即发生,从而进入条件处理部件;

$t_{\alpha} < TIME$, 表示该活动按预定时间早应发生,但因条件未满足,到目前为止实际上仍未发生,当前是否发生,仍要判断其发生的条件,即进入条件处理部件。

对于进入条件处理的活动,如果条件满足,则执行该活动子例程包括修改 t_{α} ; 如果条件不满足,则 t_{α} 保持不变。不管活动是否执行,系统仿真钟的值不作任何变动。

对所有的活动扫描一轮后,如果尚有满足条件的活动在条件处理部件中,则继续按优先级高低进行扫描,直到所有满足条件的活动均处理完,然后系统仿真钟推进到下一最早发生活动的发生时刻,并进行新一轮扫描,直到仿真结束。

(3) 进程交互法 进程交互法用“进程”的观点来抽象真实系统。它将实体历经系统时所发生的事件及活动按时间及逻辑顺序进行组合,从而形成各种进程。这种方法主要用于实体活动较规则的系

统建模。

这种策略要求建立两种事件表,第一种称为当前事件表(CEL),它存放着从当前时间点开始有资格执行的某一进程中的某一事件记录,该事件是否能发生尚未判断;第二种称为将来事件表(FEL),它包含在将来某个仿真时刻发生的某一进程中的某一事件记录。每一个事件记录中有若干个属性项,其中必须包括说明该事件所在进程及在该进程中的位置指针。

若用 $TIME$ 表示系统仿真钟的值, t_{α} 表示某一进程中的实体 α 状态发生变化的时间,进程交互法的机理可表述为:系统仿真钟推进时,首先将 $t_{\alpha} < TIME$ 的所有事件记录从 FEL 移到 CEL 中,然后对 CEL 中的每个事件记录进行扫描。它首先判断该事件所属进程及其在该进程中的位置,然后判断事件发生的条件。若条件为真,则执行该事件及后续的活动,只要可能,该进程要尽可能多地连续推进,直到该进程结束;如果条件不满足或系统仿真钟值 $TIME < t_{\alpha}$,则将该进程的事件写入 FEL 中,并退出该进程,然后对 CEL 中的下一事件记录进行处理。当 CEL 中的所有事件记录处理完后,结束对 CEL 扫描,继续推进仿真钟,即把 FEL 中最早发生的事件记录移到 CEL 中,进入新一轮的进程处理,直到仿真结束。

(4) 三阶段法 三阶段法结合了事件调度法和活动扫描法的特点,将整个仿真控制过程分为 3 个阶段,如图 1 所示。三阶段法中的活动分为 2 类,即 Bs 和 Cs。Bs 即 B 类活动,可明确预知起始和结束时间的活动。Cs 即 C 类活动,非 B 类活动即为 C 类活动。其发生时间是不可事先预知的。在三阶段法中每个实体必备的 3 个属性是:①时间片:下一状态转移时间。只有该实体属于将来某时刻发生的 B 类活动时,该属性才有意义。②可用性:是一个取布尔值的标志,用来表示将来某时刻发生 B 类活动时,该实体是否可以被无条件占用。如果标志为“真”(TRUE),则说明可用,标志为“假”(FALSE),则说明不可用。③下一活动:像“时间片”属性一样,仅当“可用性”属性为“假”时,该属性才有意义,它表示“时间片”所预期的 B 类活动。

在三阶段中的 A 阶段进行时间扫描,即扫描事件表,找出下一个最早发生事件。将系统仿真钟推进到该事件的发生时刻。系统时钟一直保持这一时刻直到下一个 A 阶段发生。注意,因为有可能有多

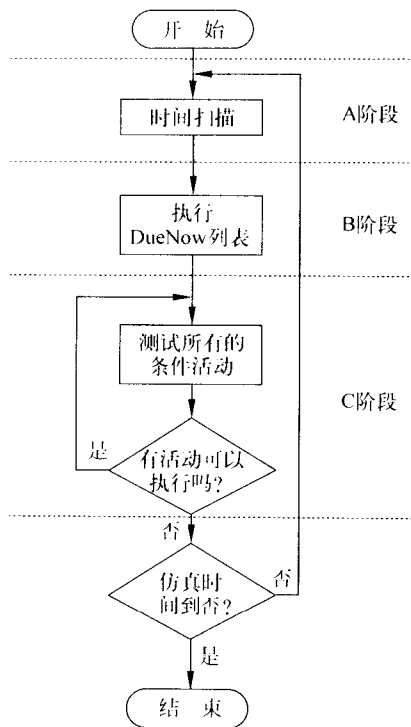


图1 三阶段法框图

个B类活动在下一时刻发生,所以仿真控制程序必须记录在该时刻所有的不可用实体而形成一个DueNow列表。

在B阶段执行DueNow列表。一旦DueNow列表形成,仿真控制程序将顺序扫描列表中的每个实体,从中挑选出可执行的实体,对于每一个可执行实体从DueNow列表中删除;将该实体的“可用性”属性置成“真”;执行该实体“下一活动”属性所代表的活动。

在C阶段,查询C类事件表。逐一对其中的事件进行条件测试,看其条件是否满足。如果条件满足,则执行相应的动作。在查询C类事件表期间,保持当前仿真钟不变,直到所有的C类事件都不满足启动条件。

参考文献

1. James W H. Strategy Related Characteristics of Discrete Event Languages and Models. Simulation, 1986, 46(4):152~159
2. Robert M D. The Three Phase Approach: A Comment on Strategy Related Characteristics of Discrete Event and Models. Simulation, 1986, 47(5):

208~211

3. 肖田元,张燕云,陈加栋. 系统仿真导论. 北京:清华大学出版社,2000 (徐文胜)

lisan shijian xitong fangzhen shuchu fenxi
离散事件系统仿真输出分析(output analysis of discrete event system simulation) 离散事件系统仿真结果的置信度分析。由于离散事件系统一般都具有随机性,每一次仿真模型运行的结果只能认为是系统模型的一次仿真抽样,因而仅从某一次仿真运行的结果来推断系统的性能并不一定能保证其结论的正确性。

基于经典统计理论的离散事件系统仿真输出分析要求恰当选择仿真运行长度,以保证仿真运行的独立性。仿真运行长度是指仿真模型中的仿真钟需要推进多少才使仿真模型执行终止。这依赖于仿真类型。

从输出分析的观点来看,离散事件系统仿真可分为两大类。

一类称为终止型仿真。它是指在规定的时间内进行仿真并统计系统性能。由于仿真运行长度是事先规定的,因而系统仿真结果与仿真运行长度有关,特别是系统的初始状态设置对仿真结果的影响不能忽略。对终止型仿真,需要多次独立运行仿真模型,以获得系统的统计性能。为提高终止型仿真各次运行的独立性,每次运行应该采用不同的伪随机数流产生随机变量,至少在每次运行时随机数发生器采用不同的种子值。这样,只有当每次运行相互独立时,每次运行所得到的仿真结果才是独立同分布的随机变量,进而才能用经典统计分析方法来构造系统性能的置信区间(置信区间就是在给定置信度下系统性能分布的范围)。

另一类称之为稳态型仿真。这类仿真的目的是要估计系统的稳态性能。它对仿真运行的长度没有限制,因而系统的初始状态设置对仿真结果的影响可以忽略。然而,为了得到系统稳态性能较好的估计值,需要确定仿真运行长度到底多长就可以认为是“足够”了,因为任何仿真都不可能是无限制地运行下去的。这与仿真结果置信区间大小的选择有关。对稳态型仿真,尽管一次“足够长”的运行可以较好地消除初始状态对仿真结果的影响,但为了估计该运行结果的置信区间,往往需要将运行过程的数据分成若干批,然后从这若干批的统计值来估计系统性能。这样,如何提高各批数据之间的独立性

直接影响置信区间的可信度。常用的方法是使每批中包含的样本数据足够多(即每批的长度足够长),且批数也足够多,以满足经典统计分析理论对统计样本的要求。

参考文献

肖田元,张燕云,陈加栋.系统仿真导论.北京:清华大学出版社,2000年6月 (肖田元 徐文胜)

lisan shuxue

离散数学 (discrete mathematics) 以离散结构为主要研究对象且与计算机科学技术密切相关的一些现代数学分支的总称。

离散数学一词始见于20世纪60年代初。随着计算机硬件和软件的迅速发展,不仅计算机的应用日益广泛和深入,而且逐渐形成了计算机科学这一独立学科及其众多的分支学科。在此过程中,曾经涉及和应用了许多现代数学学科。这些数学学科大多具有“离散性”和“能行性”两大特点。为了适应计算机科学的发展和培养教育计算机专业技术人员需要,人们就对这些数学学科中与计算机科学密切相关的内容加以分析、研究和整理,形成了所谓的“离散数学”。离散数学形成以后,发展极为迅速,现已形成了一个庞大的数学体系,应用也更加广泛。

对于离散数学的内容,人们的认识并不完全统一。目前,大多数人认为离散数学主要包括集合论、逻辑学、抽象代数、范畴论、图论、计算数论和组合学等。

参考文献

1. Hungerford T W 著.代数学.冯克勤译.长沙:湖南教育出版社,1985
2. 王兵山,王长英,周贤林,何自强编.离散数学.长沙:国防科技大学出版社,1985
3. Matt J L, Kandel A, Baker T P. Discrete Mathematics for Computer Scientists. Reston, Virginia: Reston Publishing Company, Inc., Prentice Hall Company, 1983 (王兵山)

lijie gongju

理解工具 (understanding tool) 帮助人们阅读、分析和理解软件(一般为程序)的工具。软件理解是通过各种途径分析程序代码或文档以了解程序或程序片段的功能、结构、算法以及实现细节。理解

工具是一类实用性很强、应用很广的工具。在软件开发和维护过程中,软件人员为了发现和修改程序中的错误,改进系统、扩充功能等,经常需要分析检查自己或他人编写的软件。理解工具的种类很多,典型的有:

(1) 静态分析工具 通过对程序源代码扫描和分析提取各种程序信息,包括模块调用和被调用关系、模块的控制流程结构、全局变量使用情况以及数据类型等。这些信息能以不同的图表方式显示打印,以交互方式交叉查找或做快速浏览。

(2) 源程序文档与结构图生成工具 基于对源程序代码扫描或静态分析结果,对源代码的格式进行规范化和文档化,生成模块调用与被调用关系图、模块控制流程图、PAD图、源代码结构图、对象类继承关系图等。

(3) 程序切片分析工具 把程序中与指定数据项或数据结构有关的程序部分抽取出来,滤掉与其无关的语句,并运用符号演算和逻辑推理等技术手段,帮助人们详细分析程序对指定数据项或数据结构的作用,找出执行特定路径的前置条件等。

(4) 算法模式识别工具 用以识别程序中的一些已知算法模式。算法模式识别是将人工智能和软件工程领域的技术相结合,用来帮助分析理解程序。

(5) 动态分析工具 对程序实际执行过程和执行轨迹、数据流演变过程和执行结果进行记录和分析,以帮助理解程序功能和实现细节。

(6) 反汇编及高级程序设计语言的反编译工具 将机器指令形式的程序进行反汇编,以获得其汇编代码程序,或进一步通过反编译获得其高级语言的源代码,从而提高程序的可读性。

(7) 文档分析工具 检查文档间的一致性,对文档修改后产生的波及影响进行追踪分析。

软件理解工具是计算机辅助软件工程(CASE)环境的重要组成部分,也是逆向工程和再次工程的核心工具。

参考文献

1. Beck J and Eichmann D. Program and Interface Slicing for Reverse Engineering. Proceedings, Working Conf. on Reverse Engineering. Los Alamitos: IEEE Computer Society Press, May, 1993
2. Brown P and Stafford D W. Graph Services for Program Understanding. Proceedings, Assessment of Quality Software Development Tools. Los Alamitos;

IEEE Computer Society Press, 1992

3. Cleveland L. A Program Understanding Support Environment. IBM Syst. J. 1989, 28(2):324~344

(刘超)

liti shijue

立体视觉 (stereo vision) 模仿人体双眼视觉原理,利用相隔一定距离的两个(有时可用三个)摄像镜头摄取两幅(或三幅)数字图像,通过对同一物体在两幅图像上相对位置的差异,计算获得所摄空间景物的三维信息,并由此对景物进行定位、自动识别或理解的技术总称。

立体视觉的基本原理可由图1说明:设三维空间中物体为A,左右镜头拍摄的数字图像为 I_l, I_r ,镜头相间距为 b ,焦距为 f ;那么利用物体A上某点 $p(x, y, z)$ 在 I_l 及 I_r 上的位置 $p_l(x_l, y_l)$ 及 $p_r(x_r, y_r)$,即可由公式

$$\begin{aligned}x &= b \frac{(x_l + x_r) / 2}{(x_l - x_r)} \\y &= b \frac{(y_l + y_r) / 2}{(x_l - x_r)} \\z &= b \frac{f}{(x_l - x_r)}\end{aligned}$$

计算得到有关 p 点的三维坐标值 x, y, z 。如果三维物体表面的许多点都可按上面方法求得其三维坐标,那么,有关此物体的三维信息就可以认为是得到了。

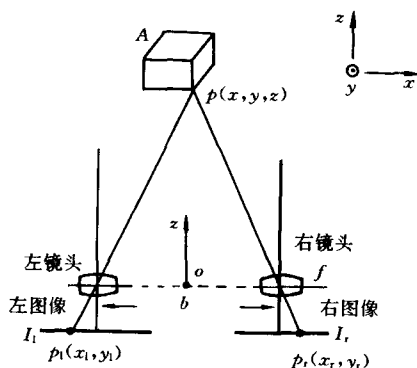


图1 立体视觉原理图

立体视觉技术中最困难的问题之一是所谓“对应点匹配”。空间点 p 在图像上的位置 p_l 及 p_r 是两个点,它们对应于同一空间点,故称为对应点。如何从一幅图像上的点 p_l (或 p_r)寻找在另一幅图像上相

对于同一物点的点 p_r (或 p_l)的技术问题,称为对应点匹配问题。

解决对应点匹配问题的方法一般可分两类。一为相关匹配方法:在待匹配的图像中选取一批候选的匹配对应点,从中选出对应点图像邻域间相关系数为最大的点,作为被匹配的对应点。二为搜索匹配,它是根据摄像机和物体的位置关系,物体本身的几何形状特征等约束条件,在两幅图像中搜索满足最优约束条件的点,作为对应点。尽管原理比较简单,但由于实际图像中需要计算的对应点的数量比较大,计算量及搜索的范围迅速增长,造成计算时间长,容易错配或失配的现象,因而对应点匹配仍是一个需要研究的问题。

除了对应点匹配之外,还可以作对应边线的匹配、对应曲面投影边的匹配。通过获得点、线、边的匹配,其他面上的点,需要通过一定的计算来获取其三维信息。

综上所述,完成立体视觉需要四个步骤,即:

- ①摄取两幅(或以上)图像;
- ②求取对应点的匹配;
- ③利用前述公式,求取对应点的三维坐标,获取物体的全面三维信息;
- ④根据三维信息对物体进行分割、建模、识别及理解。

参考文献

Horn B K P. Robot Vision. MIT Press, 1986

(袁保宗)

licheng

例程 (routine) 可多次使用的计算机程序或其一部分。这里,程序一词常指低级语言程序。也有人把例程视为子例程的同义语。

例程的概念几乎是伴随着计算机程序而产生的。大多数程序都需要将其求解问题的结果输出,这就有了输出例程。它用于启动输出设备,将数据按其规定格式送至输出设备,控制、监督输出操作等。又如,用户常常通过提示指令(命令)的执行顺序,或通过提示执行结果,对计算机程序进行检查,这可以用跟踪例程来完成。除了上面讲到的两个例程外,最常用的服务性例程还有汇编例程、编辑例程、输入例程、故障诊断例程、分类例程等。

一个较大的软件系统往往由若干个例程组成。比如,操作系统可由诸如文件管理例程、资源管理例程、作业调度例程、输入输出例程、时钟管理例程、同步出口例程、错误分析出口例程等组成。

例程有可复用例程、可再入例程、递归例

程等。

大多数例程一经装入就可执行多次,此类例程称为可复用例程。可再入例程是指本例程执行尚未完成,它又可再次进入。一个可再入例程可同时为多个计算机程序使用。在具有多道管理功能的操作系统控制下,语言编译程序应是再入式的。递归例程在程序设计中也是经常需要的。这种例程可直接或间接调用自身。在使用递归例程时,必须将使用过程中尚未用完的状态保存起来。

参考文献

孙钟秀等. 操作系统原理. 北京: 人民邮电出版社, 1980 (段祥)

lianjie bianji chengxu

连接编辑程序 (linkage editor) 把多个分别编译或汇编过的程序段组合成一个大的程序段或程序的程序。有时也称为**连接程序**。

编译程序或汇编程序产生的浮动目标程序一般由三部分组成:正文,它是目标程序的主要部分,包括指令代码和数据;外部符号(全局符号)表,包括本程序段引用的名字和被其他程序段引用的名字;浮动信息表,包括再定位所需要的有关信息。连接编辑程序扫描外部符号表,寻找所连接的程序段,根据再定位信息表解决外部引用和再定位,最终把多个正文组合成一个待装入的程序。

例如,程序段 A(图 1)和程序段 B(图 2)经连接编辑程序组合后形成程序段 C(图 3)。其中 A 定义名字 D,调用 B;B 引用 D,则 C 中引用 B 定位为引用 $l_a + 1$,对 D 的引用点定位为 $l_a + r_d$ 。

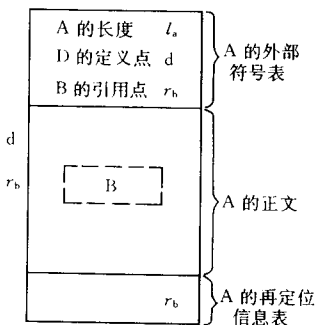


图 1 程序段 A

参考文献

Beck L L. System Software: an Introduction to

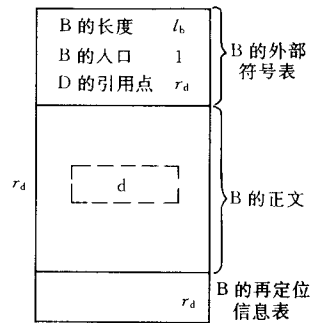


图 2 程序段 B

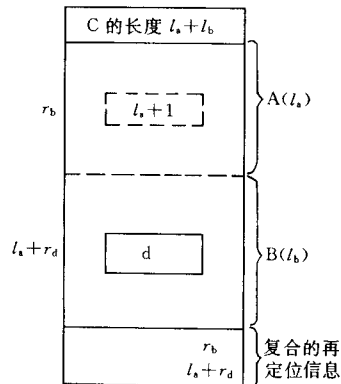


图 3 程序段 C

Systems Programming. (2nd ed). Addison - Wesley, 1990 (张素琴)

lianjie zhuanjia xitong

连接专家系统 (connectionist expert system) 一类利用人工神经网络模拟专家系统功能的系统。人工神经网络是一种连接机制的系统,故称此种专家系统为连接专家系统。其研究目的是为了改进和提高专家系统的性能。

传统的专家系统由于存在运行效率低、处理速度慢、知识获取困难等缺点,阻碍着它进一步发展。人们试图通过改进它的匹配算法、执行并行处理、增加知识获取机制,来提高它的性能。80 年代重新崛起的人工神经网络,由于具有并行处理和学习的固有特性,很快被许多专家、学者用于解决专家系统的上述问题。

主要研究内容为如何用神经网络来实现专家系统各主要组成部分的功能。下面举 3 个例子说明其实现方法。

(1) 使用人工神经网络来建造专家系统。构造替代推理机功能的神经网络连接模型和相应的知识库。例如 6 种症状、2 种疾病、3 种治疗方案的疾病诊断和治疗专家系统。选择 8 个合适的病人病历作为训练样本集,用 x_1, x_2, \dots, x_6 表示症状; x_7, x_8 表示疾病名; x_9, x_{10}, x_{11} 表示治疗方案。从病历中分别采集症状有、无或无记录三种信息;疾病的是、否或无记录三种信息;治疗方案是、否执行两种信息,它们分别以 +1, -1, 0 表示。训练样本集见表 1。

表 1 训练样本集

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}
TE1	1	1	1	-1	0	-1	1	-1	1	-1	1
TE2	-1	-1	-1	1	1	-1	-1	1	1	1	-1
TE3	-1	-1	1	1	-1	1	1	1	-1	-1	-1
TE4	1	1	-1	-1	1	-1	-1	-1	-1	-1	-1
TE5	1	-1	0	1	1	1	1	1	-1	1	1
TE6	1	-1	-1	1	1	-1	1	1	1	1	-1
TE7	1	1	1	-1	-1	1	1	-1	-1	-1	-1
TE8	-1	1	1	-1	1	1	-1	1	-1	-1	-1

全连接的神经网络经训练后构造成全连接模型,见图 1。图中连接线旁的数值为权值,神经元内的数值是附加值。把输出层中代表疾病治疗方案的 x_9, x_{10}, x_{11} 分成两个输出层,是由于学习算法的需要,插入一 x_a, x_b, x_c 附加层而形成的。神经元取值 +1, 0, -1, 特性函数为一离散型的阈值函数,计算公式为

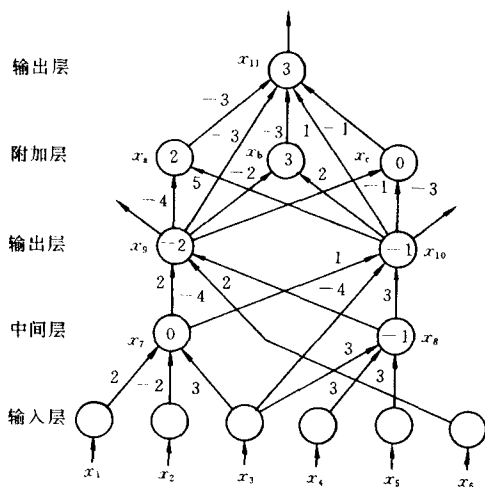


图 1 非全连接模型

$$X_j = \sum_{i=1}^n \omega_{ij} x_i$$

$$x_i = \begin{cases} +1 & \text{当 } X_j > 0 \\ 0 & \text{当 } X_j = 0 \\ -1 & \text{当 } X_j < 0 \end{cases}$$

如对网络输入病人症状: $x_1 = 1, x_2 = -1$ 和 $x_3 = -1$, 可得 $x_7 = 1$ 。因有

$$(2) \times 1 + (-2) \times (-1) + (3) \times (-1) > 0$$

如果其他症状全无(x_4, x_5, x_6 均为 -1), 类似地可得 x_9 不会出现($x_9 = -1$), 将有治疗方案 x_{10} ($x_{10} = 1$)。

上述的权值和各类症状的疾病变量名构成知识库, 网络的神经元计算代表了推理的过程。

(2) 把逻辑推理融入神经网络, 构造成一种新的神经逻辑网络(NLN)。它既能模式识别, 又能逻辑推理。这种神经逻辑网络是一个带有节点和有向连接的有向图, 见图 2。某些节点为其输出节点、输入节点, 其余为隐节点。对每一连接有一序对(a, b)与之关联, 这里 a, b 是任一实数。每一非输入节点(即输出节点和隐节点)与有序对(1, 0), (0, 1)和(0, 0)之一相关联。值(1, 0)的逻辑意义为“真”, 值(0, 1)为“假”, 值(0, 0)为“不知”。如果节点代表事物的特征, 则它们又分别表示特征“存在”, “不存在”和“不知是否存在”。

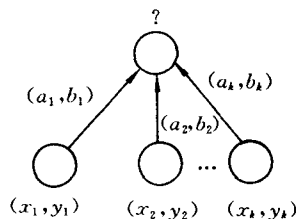


图 2 一个简单的 NLN

一旦定义了一个神经逻辑网络, 把它视作程序或函数, 从给定的一组输入去计算一组输出, 其计算过程如下:

- ① 赋给每个输入节点(1, 0), (0, 1)和(0, 0)中任一值;
- ② 用下述传播算法计算每一节点的值:

$$? = \begin{cases} (1, 0) & \text{当 } (\sum_{i=1}^k a_i x_i - \sum_{i=1}^k b_i y_i) \geq 1 \\ (0, 1) & \text{当 } (\sum_{i=1}^k a_i x_i - \sum_{i=1}^k b_i y_i) \leq -1 \\ (0, 0) & \text{其他} \end{cases}$$

③ 应用传播算法于网络每一节点,重复上述传播过程直到所有节点不再变化,即停止计算,此时网络达到稳定状态,输出节点的值是给定输入值的输出结果。

由于神经逻辑网络可模拟各种逻辑推理,因此它可用于模拟专家系统的功能。

(3) 把已开发专家系统的推理机和知识库程序通过翻译程序全部变换为有限连接的可运行的神经网络程序。这种连接专家系统可省去神经网络所需的训练时间,又获得神经网络的较好特性。

由于连接专家系统可确定计算复杂度,又可并行计算,一般较之其他类型的专家系统执行速度快,因而可用于实时处理,被国内外专家、学者所重视。

参考文献

1. 施鸿宝. 神经网络及其应用. 西安: 西安交通大学出版社, 1994

2. Hsu L S, Teh H H, Chan S C, et al. Fuzzy Logic in Connectionists' Expert Systems. IJCNN, Washington D C, 1990, 15 ~ 19 (施鸿宝)

lianxu xitong fangzhen

连续系统仿真 (continuous system simulation) 对系统状态随时间连续变化的系统进行仿真的各类活动的总称。以数字计算机为工具,针对用数学模型描述的连续系统进行仿真的活动称为连续系统数字仿真,有时也简称为连续系统仿真。

连续系统数学模型可分为: ① 集中参数系统模型,一般用常微分方程(组)描述,如各种电路系统、机械动力学系统、生态系统等; ② 分布参数系统模型,一般用偏微分方程(组)描述,如各种物理和工程领域内的“场”问题。

(1) 集中参数系统仿真

采用数字计算机对连续系统进行仿真时,核心问题是如何将模型在时间上离散化,并得到系统状态在离散时间点上的值,以代表系统的行为特性。这种离散化方法亦称之为连续系统仿真算法。经典的仿真算法是数值积分法和线性多步法,基于现代控制理论形成的仿真算法主要是离散相似法。

相似原理 连续系统仿真,从本质上是从时间、数值两个方面对原系统进行离散化。离散化以后得到的模型称为仿真模型。仿真模型是原连续模型的近似,但必须遵循“相似原理”。“相似原理”说明

如下:

设系统模型为 $\dot{y} = f(y, u, t)$, 其中 $u(t)$ 为输入变量, $y(t)$ 为系统变量。令仿真时间间隔为 h , 离散化后的输入变量为 $\hat{u}(t_k)$, 系统变量为 $\hat{y}(t_k)$, 其中 t_k 表示 $t = kh$ 。如果 $\hat{u}(t_k) \approx u(t_k)$, $\hat{y}(t_k) \approx y(t_k)$, 即 $e_u(t_k) = \hat{u}(t_k) - u(t_k) \approx 0$, $e_y(t_k) = \hat{y}(t_k) - y(t_k) \approx 0$ (对所有 $k = 0, 1, 2, \dots$), 则可认为两模型等价,这称为相似原理(参见图 1)。

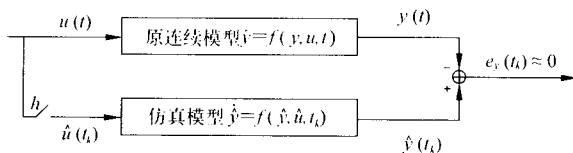


图 1 相似原理

为满足相似原理,对仿真算法有 3 个基本要求:

① 稳定性 仿真模型不改变原模型的稳定性,若原连续系统是稳定的,则离散化后得到的仿真模型也应是稳定的。

② 准确性 有不同的准确性评价准则,最基本的准则是:

绝对误差准则: $|e_y(t_k)| = |\hat{y}(t_k) - y(t_k)| \leq \delta$

相对误差准则: $|e_y(t_k)| = \left| \frac{\hat{y}(t_k) - y(t_k)}{\hat{y}(t_k)} \right| \leq \delta$

其中, δ 表示规定的误差量,这决定于仿真要求。

③ 快速性 数字仿真是一步一步推进的,即由某一初始值 $y(t_0)$ 出发,逐步计算,得到 $y(t_1)$, $y(t_2)$, \dots , $y(t_k)$, 每一步计算所需时间决定了仿真速度。如果第 k 步计算对应的系统时间间隔为 $h_k = t_{k+1} - t_k$, 计算机由 $y(t_k)$ 计算 $y(t_{k+1})$ 需要的时间为 T_k , 那么,若 $T_k = h_k$, 称为实时仿真,若 $T_k < h_k$ 称为超实时仿真,而大多数情况是 $T_k > h_k$, 对应于离线仿真。

上述三个方面,其中稳定性是必须保证的,而准确性和快速性依赖于仿真要求,但准确性和快速性两者是互为矛盾的,往往需要根据仿真要求进行折衷。

数值积分法仿真 数值积分法仿真是用数值计算的方法对用微分方程描述的系统进行仿真。

在许多科学技术领域中,常常采用微分方程描述系统的模型,包括常微分方程和偏微分方程。传统的数学分析方法只能解决少数比较简单或典型的微分方程的求解问题,而对于比较复杂的如高阶微

分方程、变系数微分方程、非线性微分方程、一般的偏微分方程等要得到其解析解是非常困难的,甚至是不可能的。为了实现微分方程的积分,求解这些方程,数值方法被广泛采用,以获得其近似解。

线性多步法仿真 线性多步法仿真是数值积分法仿真的一种。用数值计算的方法对用微分方程描述的系统进行仿真计算时;如果仅用到前一步计算得到的数据,称为单步法;如果采用前若干步计算得到的数据,称为多步法。

多步法仿真的每一步计算不需要多次计算微分方程的右端函数值,计算量比单步法少得多,因而速度快。但它在某些情况下需要单步法的帮助。另外,它还要求在整个仿真过程中仿真步长保持不变。

离散相似法仿真 离散相似法仿真是基于采样定理将连续系统进行离散化处理,求得与它等价的离散模型的仿真方法。

由于连续系统的模型可以用传递函数来表示,也可以用状态空间模型来表示,因此,与连续系统等价的离散模型可以通过两个途径获得,其一是基于状态方程离散化,得到时域离散相似模型;其二是对传递函数作离散化处理得到离散传递函数,称为频域离散相似模型。

此外,还有诸如面向结构图仿真、根匹配法仿真、可调整积分法仿真和病态系统仿真等多种仿真方法。

在某些情况下,连续系统的模型(或部分模型,如采样控制系统的数字控制器)采用差分方程描述,对这类离散时间模型则无需再进行离散化处理,但一般将其归类为连续系统仿真范畴。

(2) 分布参数系统仿真

是面向用偏微分方程描述的一类系统的仿真。分布参数系统的运动规律一般采用偏微分方程来描述。对于确定型的偏微分方程,采用一阶描述形式,可以用以下表达式:

$$F_0(\phi, p, z, t) \frac{\partial \phi}{\partial t} + \sum_{i=1}^k F_i(\phi, p, z, t) \frac{\partial \phi}{\partial z_i} = f(\phi, p, u, z, t) \quad (1)$$

$$b(\phi, p, z, t) = 0 \quad z \in \delta_z \quad (2)$$

$$\phi(z, 0) = \phi^0(z) \quad (3)$$

$$y(z, t) = g(\phi, p, z, t) \quad (4)$$

$$h(\phi, p, u, z, t) \geq 0 \quad (5)$$

上述表达式中的有关符号说明如下:

① 自变量:常微分方程中自变量只有时间变量

t ,而在偏微分方程中,除了时间 $t \in T$ 以外,还有空间自变量 $z \in Z$;

② 输入变量 $u \in U$ 及输入段集合:映射 $Z \times T \rightarrow U: u(z, t)$;

③ 因变量: $\phi \in \Phi$, 且是 z 与 t 的函数;

④ 式(2)确定边界条件, δ_z 表示 Z 的边界;在 $z \in \delta_z$ 上, ϕ 随时间变化满足该等式;

⑤ 式(3)表示初始条件,即规定初始时刻 ϕ 在域内的值;

⑥ 输出变量 $y \in Y$ 是空间和时间的函数;

⑦ 式(5)规定了约束条件。

在某些情况下,系统是以高阶偏微分方程的形式给出。一般说来,经过适当变换,高阶偏微分方程可以转换成一阶偏微分方程组。

分布参数系统仿真方法主要有3类,一类是经典方法——差分法,第二类是线上求解法,第三类是现代方法——有限单元法。

为了使仿真模型能在计算机上运行,必须将仿真模型转换成程序。同时程序还必须具有交互能力,以便进行仿真实验。仿真技术发展的重要标志之一是仿真程序水平不断提高,发展到仿真语言(参见仿真语言)阶段。

参考文献

1. Vlach J, Singhal K 著. 电路分析和设计的计算机方法. 汪蕙等译. 北京:科学出版社,1992

2. 肖田元,张燕云,陈加栋. 系统仿真导论. 北京:清华大学出版社,2000 (肖田元)

lianji fenxi chuli

联机分析处理 (online analytical processing, OLAP) 以数据仓库中海量数据为基础的联机的复杂分析技术和过程。OLAP 支持人们从不同的角度快速灵活地对数据仓库中的海量数据进行联机的复杂查询和多维分析处理。

联机分析处理 (OLAP) 概念是 E. F. Codd 于 1993 年提出的。鉴于这位关系数据库创始人的影响,OLAP 技术受到广泛重视,促进了 OLAP 技术的发展,并使 OLAP 发展成为与**联机事务处理 (OLTP)** 明显区分的一类计算机应用。OLAP 软件提供对数据的多维分析功能。但若要进行更深层次的分析,发现数据中隐含的规律和知识,则需要**数据挖掘技术**和相应的数据挖掘软件来完成。

多维数据模型是面向分析的数据模型,是数据分析时用户的数据视图。多维数据模型给分析人员

提供多种观察数据的视角和面向分析的操作。

多维数据模型的**数据结构**可以用一个多维数组来表示,即(维1,维2,...,维 n ,度量值),其中维是人们观察数据的特定角度。例如图1中所示的商品销售数据是按时间、地区、商品种类,加上变量“销售额”组成的一个多维数组(地区,时间,商品种类,销售额)。它有三个维:地区维、时间维、商品种类

维,销售额是度量值。人们可以观察销售额随时间、地区、商品种类及其各种组合的变化情况。维还可能存在细节程度不同的多个描述方面,称之为维的层次。例如年、季、月、日等就是时间维的一种层次;同样,县、市、省、国家等构成了地区维的一种维层次。

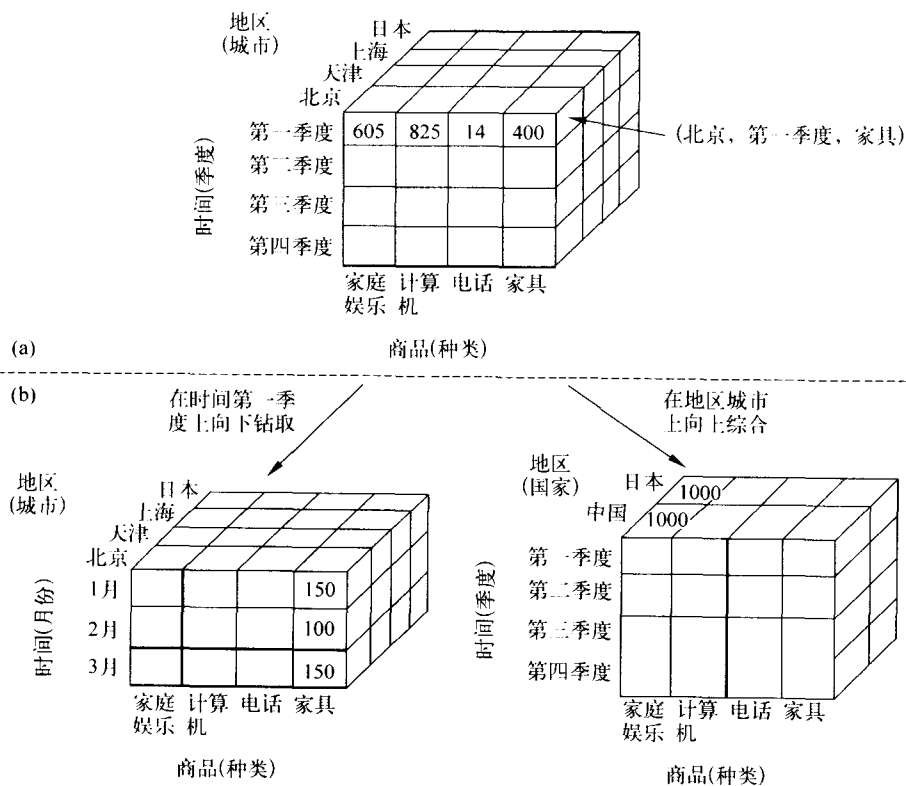


图1 商品销售的多维数据模型

在多维数据模型上基本的 OLAP 多维分析操作有切片、切块、旋转、向上综合、向下钻取等。通过这些操作,使用户能从多个角度多个层次观察数据、剖析数据,从而深入地了解包含在数据中的信息与内涵。

联机分析处理(OLAP)服务器软件透明地为上层分析工具软件 and 用户提供多维数据视图和多维存取接口。分析软件 and 用户不必关心它的分析数据(即多维数据)到底储存在何处,是如何储存的。OLAP 服务器软件则必须考虑多维数据模型的实现技术,包括如何组织多维数据,如何储存多维数据,多维数据的索引技术,多维查询语言的实现(语言

的语法分析、编译、执行和结果表示),多维查询的优化技术等。

联机分析处理(OLAP)服务器软件一般按照多维数据模型的不同实现方式,有多维联机分析处理(MOLAP)结构、关系联机分析处理(ROLAP)结构和混合联机分析处理(HOLAP)结构等多种结构。

MOLAP 结构直接以多维立方体(CUBE)来组织数据,以多维数组来储存数据,支持直接对多维数据的各种操作。人们称这种按照多维立方体来组织和储存的数据集为多维数据库(MDDB)。

ROLAP 结构用关系数据库管理系统(RDBMS)来管理多维数据,用关系(二维)表来组织和储存多

维数据,将多维立方体上的操作映射为标准的关系操作。ROLAP 将多维数据划分为两类,一类是事实表,另一类是维表。事实表用来描述和储存多维立方体的度量值及各个维的码值;维表用来描述维信息,包括维的层次及成员类别等。也就是说,ROLAP 用“星状模式”和“雪片模式”来表示多维数据模型。星状模式通常由一个中心表(事实表)和一组维表组成。星状模式的事实表与所有的维表相连,而每一个维表只与事实表相连。维表与事实表的连接是通过码来体现的。维通常有层次,对维表按层次进一步细化后形成的星状模式的变形称为“雪片模式”。

HOLAP 结构将 ROLAP 和 MOLAP 结合起来,例如,将细节数据存在关系数据库中,而将综合数据存在 MOLAP 服务器中,既利用了 ROLAP 可扩展性好的优点,也利用了 MOLAP 计算速度快的优点。

E. F. Codd 在 1993 年提出了关于 OLAP 的 12 条准则,系统阐述了有关 OLAP 数据分析模型的一系列概念,OLAP 软件产品的特点和衡量标准,这对 OLAP 产品的识别及发展方向都起了重要作用。现在 OLAP 分析工具及软件产品已在商业数据库领域得到广泛应用。

参考文献

1. 萨师煊,王珊. 数据库系统概论(第三版). 北京:高等教育出版社,2000
2. 王珊等. 数据仓库技术和联机分析处理. 北京:科学出版社,1998
3. Codd E F, Codd S B, Salley C T. Providing OLAP(Online Analytical Processing) to User-Analysts. PC WORLD, 1993

(王珊)

lianji shiwu chuli

联机事务处理(online transaction processing, OLTP)

一种以大量、简单、可重复使用的例行短事务为主的数据库应用方式。可对数据库进行即时查询、更新或其他操作,并使数据库中的数据总是保持在最新的一致状态。在数据处理过程中,一般将一个数据库操作序列指定为一个事务。事务具有下列 4 个性质:①执行的原子性 事务在执行时,应遵守“要么不做,要么全做”的原则,即不允许事务部分地完成。即使因为故障而使事务未能完成,在恢复时也要消除其对数据库的影响。②功能上保持一致性 即事务的执行应使数据库从一个一致状态转变到另一个一致状态。③事务间的隔离性

如果多个事务并发地执行,应像各个事务独立执行一样,并发控制可以保证事务间的隔离性。④作用的持久性 一个成功执行的事务对数据库的影响应是持久的,即使数据库因故障而受到破坏,也是如此。

联机事务处理系统是一种信息系统。联机事务处理以交互方式为用户提供服务。用户通过终端、微型计算机或其他设备(如自动取款机)输入事务,经系统处理后回送结果。联机事务处理系统一般需要高性能计算机网络和分布式数据库管理系统支持。联机事务处理系统已有广泛的应用,例如,民航售票、银行出纳、证券交易、超市销售等都是它的典型应用实例。

(孙志挥)

lianji shouxie hanzi shibie

联机手写汉字识别(online handwritten Chinese characters recognition)

用一种专门的笔在图形输入板(或个人数字助理 PDA 的具有压感膜的液晶屏幕)上写字,将字的笔画轨迹、书写顺序和压力等信息输入计算机中,人一面写,机器一面辨认所写汉字的过程和技术。

计算机从书写的笔画轨迹中提取笔画、笔顺和笔画结构信息,获得汉字的笔画结构描述。利用待识汉字的笔画结构描述与标准笔画结构描述进行匹配比较,按其异同差异大小决定识别结果。由于实时笔画信息的获取,使联机汉字识别的困难程度比脱机识别大为减小(参见脱机手写汉字识别)。

若在联机手写汉字识别中利用笔顺信息,则要求书写者的笔顺固定,这样可以极大地简化汉字的结构匹配识别过程;对于无笔顺书写限制的汉字识别,则只能利用书写笔画的相关位置信息进行模式匹配。但是由于不同人书写笔画轨迹的不同,对笔画分割和类型判决带来了困难,使得一般的笔画结构匹配算法遇到较大困难。利用笔画结构的隐马尔可夫模型(HMM)及笔画信息全局统计识别的方法,即在已知笔迹信息条件下的统计识别算法,对无笔顺联机汉字识别取得了重要的进展。对于各种草书汉字的识别问题,在获得足够样本的条件下,利用已知笔画信息的全局统计识别算法,是较容易解决的问题。

目前,联机手写汉字识别已经推广应用,书写比较规则时识别率可达 98%,特别是以嵌入方式在个人数字助理(PDA)和手机中得到了很好的应用。

(丁晓青)

lianxiang cunchuqi

联想存储器 (associative memory) 根据给定内容的特征而不是根据地址进行存取的存储器。又称关联存储器,或按内容寻址存储器。这种存储器与一般存储器不同之处在于除了有一般存储器存储信息的功能外,还有对信息进行处理的功能。

访问联想存储器时,将给定的数据特征(或称比较数据)与存储单元中的全部或部分数据进行比较,若符合,则将该存储单元的全部或部分内容按要求读出,或将新的数据写入该存储单元。当有多个存储单元都具有相同的数据特征时,就是多重符合。对于写,可将新的数据都写入这些符合的存储单元。

联想存储器框图如图1所示。它由存储单元阵列和一些逻辑电路组成。存储单元阵列中的每个存储单元都有能完成存储、比较、读写、控制等功能的逻辑电路。比较数据寄存器存放要查找的数据。用这个数据的全部或一部分作为数据特征,采用哪一部分可由屏蔽寄存器来决定。比较数据寄存器和屏蔽寄存器是逐位对应的。屏蔽寄存器的某一位为“1”,则对应于该位的比较数据寄存器中的那一位作为要查找的特征;反之,如果屏蔽寄存器的某一位是“0”,则数据寄存器中的相应的信息位被屏蔽。例如:

比较数据寄存器的内容: 1 0 1 1 0 0 1 0

屏蔽寄存器的内容: 1 1 1 0 0 0 0 0

则查找的特征是 1 0 1 × × × × × (“×”是任意,即“1”或“0”)。如果有两个存储单元,它们存放的信息如下:

存储单元1的内容: 1 0 0 1 1 1 0 1

存储单元2的内容: 1 0 1 1 1 0 1 1

查找的结果是存储单元2的内容符合。

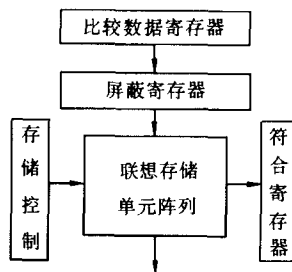


图1 联想存储器框图

符合寄存器中的每一位对应存储单元阵列中的

1个存储单元。查找是并行进行的。当某个存储单元的内容符合,则对应的符合寄存器那一位为“1”,否则,为“0”。当符合寄存器的内容不止是1个“1”时,就发生了多重符合。

以上访问过程都是用电路完成的。按符合寄存器里“1”所对应的存储单元进行读出或写入,就完成了对联想存储器的访问。

联想存储器可实现存储器的并行操作,特别适合于信息检索和更新。

联想存储器的信息处理功能可以有多种比较操作,如全等、不等、小于、大于等。联想存储器比一般存储器复杂,造价也高,所以一般采用的联想存储器的规模都不大。(徐炜民)

lianxiang jiyi

联想记忆 (associative memory) 其输入能特定唤起所联想的响应的记忆系统。人脑的记忆就是以联想的方式工作的。联想记忆的作用可由图1表示,它执行一种由输入向量 X 联想映射为输出向量 V 的变换,即

$$V = M[X], \quad X \in R^n, V \in R^m \quad (1)$$

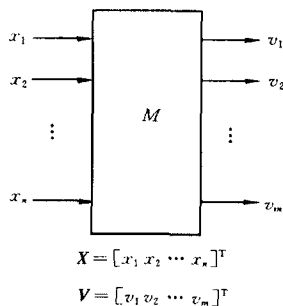


图1 联想记忆框图

算子 M 表示一般的非线性矩阵型算子。随记忆模型不同而有不同的形式。 M 的初值通常由给定的待存储的原型向量(样本)表达。 M 的值应满足如下条件:输入向量 X 后,输出是所存向量中与 X 最近的向量(或它应联想的向量)。计算 M 的算法称为记录或学习算法。而由一个包含部分输入信息的向量 X 通过式(1)进行的映射称为恢复或回忆。假定有 p 个如下的联想的存储向量对

$$x_i \rightarrow v_i \quad i = 1, 2, \cdots, p \quad (2)$$

且 $v_i \neq x_i$, 则网络称为异联想记忆,若 $v_i = x_i$, 则称为自联想记忆。

与通常的计算机中按地址存储信息的方式不同,联想记忆是把信息分布存储在算子 M (在神经网络中对应于连接权矩阵) 中。可以直接由信息的内容去回忆它,因而是一种按内容存储方式。

按回忆方式不同可把联想记忆网络分为静态记忆和动态记忆网络。前者执行的是一种对输入的前向映射。动态网络的记忆过程是输入和输出的交互反馈作用。网络经演变收敛于一个平衡点(一般是网络的不动点吸引子),就是回忆的结果。由于动态网络有较好的容错性,所以是目前最常用的一种,常见的动态记忆网络是 **Hopfield 神经网络模型**(用于自联想记忆)和 **Kosko 的双向联想记忆模型**(用于异联想记忆)。

从应用观点看,对联想记忆的主要要求一是希望它能存的样本多,即容量大,二是每个样本有较大的容错性,这样才能从不完整的或畸变了样本恢复它的原型。增大网络规模对上述两个性能是有好处的,但这样做降低了资源利用率,因而通常用允许存储的样本数与网络中神经元总数的比值来表示容量。对给定的网络来说,提高容量与增强容错性往往是矛盾的,研究更好的学习算法和网络结构以达到较大容量并同时保证高的容错能力是很重要的课题。

联想记忆在模式识别、图像恢复、智能控制、优化计算以及光学信息处理等领域有很广泛的应用前景,因而在人工神经网络和人工智能研究中都占有很重要的地位。

参考文献

1. Kohonen T. Self-Organization and Associative Memory. Berlin: Springer-Verlag, 1984
2. Zurada J M. Introduction to Artificial Neural Systems. New York: West Publishing Company, 1992
3. 焦李成. 神经网络系统理论. 西安: 西安电子科技大学出版社, 1990 (阎平凡 王正欧)

liangzi jisuan

量子计算 (quantum computing) 基于量子力学理论和量子器件的计算方式。量子器件是以量子效应为工作基础的器件,由量子器件构成的计算机称为量子计算机。

R. P. Feynman 于 1982 年指出:按照量子力学原则建造的新型计算机对解某些问题可能比常规计算机更有效。1985 年, D. Deutsch 指出:利用量子态的相干叠加性可以实现并行的量子计算。1994 年, P.

W. Shor 提出一种量子算法来解密码学中的大数因子分解问题。这是一个用传统计算机甚至超级计算机都几乎不能解决的复杂性问题,但是用 Shor 量子算法就能够在多项式时间内得到有效解。1996 年, L. K. Grover 提出量子搜索算法。在 N 个元素的集合中搜寻某个元素,经典算法搜寻 $N/2$ 次后,找到的概率为 $1/2$ 。Grover 量子搜索算法则只需 $N^{1/2}$ 次即可达到同样的概率。当 N 很大时, Grover 量子搜索算法很有效。

在传统计算机中,信息单元用二进制的的一个位来表示,它不是处于“0”态就是处于“1”态。在二进制量子计算机中,信息单元称为量子位,它除了处于“0”态或“1”态外,还可处于叠加态。叠加态是“0”态和“1”态的任意线性叠加,它既可以是“0”态又可以是“1”态,“0”态和“1”态各以一定的概率同时存在。通过测量或与其他物体发生相互作用而呈现出“0”态或“1”态。任何两态的量子系统都可用来实现量子位,例如氢原子中的电子的基态和第一激发态、质子自旋在任意方向的 $+1/2$ 分量和 $-1/2$ 分量、圆偏振光的左旋和右旋等。

一个氢原子可用于存储一个量子位,当它处于基态时表示“0”,处于激发态时表示“1”。如果要从氢原子中读取(或者写入)量子信息,可使用激光脉冲。当脉冲中的光子具有基态与激发态之间的能量差时,氢原子中的电子将从一个状态跳转到另一个状态。

n 个量子位的有序集合称为 n 位量子寄存器。处于叠加态的 n 位量子寄存器中的数是从 0 到 $2^n - 1$ 的所有数,它们各以一定的概率同时存在。在传统计算机中,一个 n 位寄存器只能保存一个 n 位二进制数;而在量子计算机中,一个 n 位量子寄存器可以同时保存 2^n 个 n 位二进制数。而且,量子计算机还可以对这些数同时进行运算。量子寄存器位数的线性增长使存储空间指数增长以及量子并行是量子计算机的特性。

对量子位的态进行变换,可以实现某些逻辑功能。变换所起的作用可以相当于传统计算机中的逻辑门所起的作用。在一定的时间间隔内实现逻辑变换的量子装置称为量子逻辑门。量子逻辑门包括量子“非”门、量子“与”门、量子“异或”门等。量子逻辑门可以组成量子半加器、量子全加器、多位量子加法器等。多个量子逻辑门相互连接起来可以构成一台量子计算机。

量子计算机的计算过程由量子算法决定,不同

的算法有不同的变换。计算过程可由传统计算机控制。量子并行是量子计算机的特点,然而对于串行计算及迭代运算,量子计算机不具备优势。因此量子计算机适合于作为传统的通用电子计算机的高速协处理器或外围专用处理机,或专门为实现某种量子算法或模拟某种量子系统的专用计算机。

量子计算的最困难之处是它的物理实现。在实验室中已实现了少量的量子位,而一台有实际应用价值的量子计算机往往需要 10 万甚至更多个量子位。另外,量子计算机的计算时间也受到很大的限制。量子位中保持激发态的时间大约仅 1 ms,而激光脉冲持续约 1 ns,因此,1 次量子计算仅能进行大约 1 000 次逻辑操作。研制量子计算机还需要在物理实现方面解决很多问题,例如如何使量子系统和环境隔离以维持量子逻辑的一致性、如何使量子位的初始化方便而迅速(即可以随时设置量子位的初始态)以及如何解决量子器件之间的连接等。虽然对量子计算机的物理实现已进行了不少研究工作,但研制有实用价值的量子计算机还非常困难。

量子计算是一种和传统的计算方式迥然不同的新型计算方式。对于某些问题,量子计算机可以达到传统计算机不能达到的解题速度,可以解决传统计算机不能解决的某些问题。量子计算仅处于起步阶段,如果以后能研制出有实用价值的量子计算机,就可以使人类更好地探索自然和驾驭自然。

参考文献

1. Steane A M and Rieffel E G. Beyond Bits: The Future of Quantum Information Processing. Computer, Jan. 2000, 38 ~ 45
2. 夏培肃. 量子计算. 计算机研究与发展, 2002, 1 (张中)

lingyu gongcheng

领域工程(domain engineering) 针对一组具有相似或相近需求的应用系统建立复用基础设施的**软件工程**。这里领域指一组具有相似或相近软件需求的应用系统所覆盖的功能区域。领域工程覆盖了建立软件的可复用构件的所有活动,是一种系统化的**软件复用技术**。

领域工程对领域中的多个应用系统进行分析,标识这些应用的共同特征和可变特征,对刻画这些特征的对象和操作进行选择 and 抽象,形成领域分析模型。然后,依据领域分析模型及应用体系结构形成领域中应用共同具有的特定领域软件体系结构

(DSSA)或应用生成过程,并以此为基础识别、开发和组织可复用构件。这样,当开发同一领域中的新应用时,可以根据领域分析模型,确定应用的需求规约,根据领域特定的软件体系结构形成应用的设计,并以此为基础选择可复用构件进行组装,从而形成目标系统。对于一些特定的领域,在工具的有力支持下,甚至可以由新应用的需求规约通过变换直接生成系统。

领域工程分为三个主要阶段,即领域分析、领域设计和领域实现。通常,这三个阶段是顺序进行的。同时,领域工程是反复迭代、逐步精化的过程,在实施领域工程的每个阶段中,都可能返回到以前的步骤,对以前的步骤得到的结果进行修改和完善,再回到当前步骤,在新的基础上进行本阶段的工作。

(1) 领域分析 本阶段的主要目标是获得领域分析模型。在本阶段之前需要进行一些准备工作,例如:确定领域工程的目标,制定领域工程的规划,定义领域的边界,识别信息源等。以此为基础,可以进一步分析领域中应用系统的需求,确定哪些需求是被领域中的应用系统广泛共享的,哪些需求是特定应用系统所独有的,在什么情况下具有这些需求,以及这些需求之间的关系,例如:多选一关系、成组可选关系、依赖关系等。当领域中存在大量系统时,需要选择它们的一个子集作为样本系统。

(2) 领域设计 本阶段的主要目标是获得 DSSA。由于领域分析模型中的领域需求具有一定的变动性,DSSA 也要相应地具有变动性。它可以通过表示具有变动性的解决方案等来达到这一点,许多现有的设计模式对于这种变化性都有很好的支持。由于复用基础设施是依据领域分析模型和 DSSA 来组织的,因此在本阶段通过获得 DSSA,也就同时形成了复用基础设施的接口和交互规约。

(3) 领域实现 本阶段的主要目标是依据领域分析模型和 DSSA 开发和组织可复用信息。这些可复用信息可能是从现有系统中提取得到,也可能需要通过新的开发得到。本阶段也可以看作复用基础设施的实现阶段。

领域工程是一项需要投入大量人力和资源的活动。但如果实施得有效,其后的应用系统可以基于复用基础设施进行开发,在保证质量的同时,大大降低这些系统的开发成本,从而使软件企业得到较高的回报。

参考文献

1. Guillermo Arango, Ruben Prieto-Diaz. Domain

Analysis Concepts and Research Directions. In Domain Analysis and Software System Modeling, R. Prieto-Diaz and G. Arango, eds., IEEE Computer Society Press, Los Alamitos, California, 1991:9 ~ 32

2. Hassan Gomaa. An Object-Oriented Domain Analysis and Modeling Method for Software Reuse. In: Proceedings of the Hawaii International Conference on System Science, Jan. . 1992:46 ~ 56 (王千祥)

liulanqi

浏览器 (browser) 作为观察网络的窗口以及在网上进行各种操作的**万维网 (WWW)** 客户端软件。浏览器可根据用户的请求, 获取万维网 (WWW) 服务器上**超文本置标语言 (HTML)** 编写的文档, 解释这个 HTML 文档, 并将文档内容以用户环境所许可的效果最大限度地显示出来。

当用户选择一个超文本链接时, 浏览器通过超文本链接相连的统一资源定位器 (URL) 来请求获取文档, 等待服务器发送文档, 然后处理这个文档, 并在客户端显示出来。

最初, 浏览器只是观察万维网的基本工具, 现在已将各种 Internet 的访问集成在一起, 包括电子邮件、新闻、Internet 电话、网上交谈、主页编辑以及多媒体游戏等。因此, 浏览器已成为重要的 Internet 软件。

目前有适合不同平台、操作系统以及图形用户界面的万维网浏览器, 它们大致可分为两类: 线模式的和图形界面的。常用的浏览器有 Lynx, Mosaic, midas WWW, Netscape Navigator 和 Microsoft Internet Explorer, 其中 Lynx 是线模式浏览器。编写 HTML 文档时, 最好同时具备有多种万维网浏览器, 以便测试 HTML 文档在不同环境下的不同的显示效果。

参考文献

1. 胡道元. 计算机网络 (高级). 北京: 清华大学出版社, 1999
2. Douglas E Comer. Internetworking with TCP/

IP, Vol. I , 3rd ed. Prentice Hall Inc. , 1995

(胡道元)

liulanqi - wanweiwang - shujuku moshi

浏览器-万维网-数据库模式 (browser - Web - database mode) 一种三层客户-服务器模式。又称 BWD 模式 (参见**客户-服务器计算**)。在 BWD 模式中, 浏览器是客户, 数据库端是服务器, 而万维网 (Web) 扮演着应用服务器的角色。

这种三层计算模式的优点是提供统一的用户界面, 可以支持各种计算机、各种操作系统、各种数据库管理系统以及各种用户界面。

这种三层计算模式能提供功能性数据库服务器管理, 可以优化数据库服务器的存取管理, 并且这种优化与具体的数据库管理语言无关。作为中间层的 Web 服务器完成过程管理功能, 为客户提供与数据库服务器无关的统一界面。由于应用逻辑都集中在中间层上, 使得对这些逻辑的修改和管理的复杂性减少。而对于两层的客户-服务器计算模式, 对应用逻辑的任何修改都导致对每个客户应用的修改。

BWD 模式的另一个优点是对事务的可靠控制。由中间层 Web 管理分布式数据库的事务, 通过名字而不是通过位置来访问资源, 因此提供了更大的伸缩性和可扩展性。

在 BWD 模式中**文本数据库**存储现存的文本, 且能转换到 HTML 格式 (参见**超文本置标语言**), 并在 Web 浏览器上显示。通用网关接口 (CGI) 是数据库服务与 HTML 文件之间的接口程序, 负责处理 HTML 文件运行在数据库服务器中的非 HTML 程序之间的数据交换。当用户从浏览器输入查询信息或传送数据后, 便激活一个 CGI 程序。该 CGI 程序又可调用操作系统下的其他程序, 完成用户的查询任务, 并将查询结果传给 CGI, 通过 CGI 传给 Web 服务器, 其交互流程如图 1 所示。

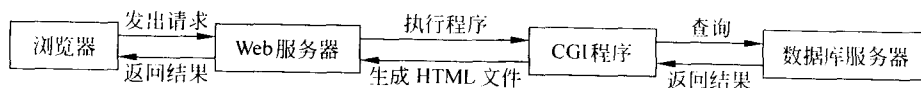


图 1 BWD 模式中 CGI 与服务器交互的流程

参考文献

胡道元. 计算机局域网 (第三版). 北京: 清华

大学出版社, 2002

(王勇 相士俊)

liukong

流控 (flow control) 确保发送实体不使接收实体发生数据溢出现象的控制机制。接收实体一般都为接收数据设置一定容量的数据缓冲器。当接收到数据后,接收器往往要对数据进行某些处理,然后再将其送给高层软件。如果不进行流控,那么,在处理已到达的数据的过程中,一旦又来了新的数据,就会发生接收缓冲器溢出的情况。

常用的流控技术有停-等流控和滑动窗口流控两种。

停-等流控 停-等流控是一种最简单的流控技术,其原理如下:发送实体发送一帧后,必须停止发送并等待接收到该帧的目的实体发回来一个确认后,才能接着发送下一帧。这样,目的实体只要控制确认的发送就可进行流控。这种技术仅适用于只有少量的大帧的情况。通常发送实体都把大的数据块分成多个小块并发送许多帧。因为,帧越大,所需的接收缓冲器就越大,传输时间也越长,就越容易出错,出错后需重发这整个一帧。这样,一个站占用介质的时间就长,其他站的发送延迟也就越长。为了避免上述问题,数据包分成许多小帧发送,那么停-等流控就不适用了。因为它在同一时刻只能允许有一帧在链路上传输,会造成链路的极大浪费。特别是当数据速率很高或者发送站与接收站之间的距离很大时,链路的利用率将很低。

滑动窗口流控 停-等流控的缺点是同一时刻只有一帧在传输。如果在同一时刻,允许有多个帧在传输,则可大大提高利用率。

假设 A 和 B 两个站以全双工(参见串行传输)链路相连接。B 站的缓冲器容量为 n 帧。A 可以发送 n 帧而不必等待确认。每个帧都有一个顺序号。作为对某一帧的响应,B 发送一个包含有下一个准备接收的帧的顺序号的响应帧。这个顺序号同时意味着 B 准备接收下一个 n 帧的起始顺序号。一个响应也可以作为对多个帧的响应。例如,B 接收到第 2、3、4 帧以后,再发送一个顺序号为 5 的响应,它表示可以接收第 5 帧开始的 n 帧,同时对 2、3、4 帧响应。A 维护一张允许发送的顺序号的表格,B 则维护一张准备接收的顺序号的表格。每张表格都可看作为帧的窗口。这种操作称为滑动窗口流控。

由于所用的顺序号占据帧中的一个域,域的位数决定了顺序号的多少。如果域占据了三位,那么帧的顺序号可以从 0 到 7。7 以后的顺序号又是 0。

一般来说,若占据了 k 位,则顺序号从 0 至 $2^k - 1$,帧的顺序号以 2^k 为模数。

若 A、B 两站都能发送和接收数据,则各自需维护两个窗口,一个用于发送,一个用于接收。为了提高效率,当双方交换数据时,可采用捎带响应的方法,即在数据帧中除了用一个域表示帧的顺序号外,另有一个域表示响应顺序号。这样数据和响应的发送可以组合在一个帧内,节省了通信容量。若站没有数据要发送,它仍然可以发送单独的响应帧。若站只有数据要发送而没有新的响应要发送,那它必须在数据帧中重发最后一个响应,因为用作响应的域必须要填入数据,而接收到重复响应的站会把这个响应甩掉。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999
2. Stallings W. Data and Computer Communications. Prentice Hall, 1997 (胡道元)

liumeiti

流媒体 (streaming media) 以流的形式在网络(互联网、无线移动网络等)上进行数字媒体(主要指音频和视频)传输的技术。它将视频、音频之类的连续媒体经压缩编码、数据打包后按照一定的时间间隔要求连续地发送给接收方,接收方在后续数据不断到达的同时对接收到的数据进行重组、解码和播放。

过去,多媒体文件(节目)需要从服务器上下载到终端设备后才能播放,这限制了人们在计算机和互联网上使用多媒体数据进行实时的交流。流媒体技术能够较好地解决这个问题。它的主要特点是边下载边欣赏,从而使用户能够不间断地在线欣赏多媒体节目。

流媒体技术所研究的主要内容包括:

流媒体编解码技术 典型的流媒体编解码技术有用于 64 kb/s 视频传输的 H. 261,面向 1.5 Mb/s 数字视频音频传输和存储的 MPEG-1,面向高品质数字视频音频传输和存储的 MPEG-2,面向交互应用和网络传输的 MPEG-4 以及适于低码率视频编码的 H. 263。最新的发展趋势是可扩展性编码,如细粒度可扩展(FGS)编解码技术等。

流媒体存储和调度策略 网络带宽和视频服务器的输入、输出往往是制约流媒体服务性能的瓶颈。在大规模点播电视(VOD)系统中,用户对媒体数据

的点播往往集中于少数热门节目,流媒体存储和调度策略的关键是合并用户服务,共享服务器和网络带宽资源。

流媒体的传输与控制 解决媒体流在两个端系统间传输的相关问题,包括媒体流拥塞控制策略、差错控制策略、速率调节策略等,其目标是提高流媒体应用的服务质量(QoS)。

多媒体代理服务器及内容替换机制 多媒体代理服务器将一些访问频繁的多媒体数据存储在内存或硬盘中,当用户通过多媒体代理服务器访问这些数据时,多媒体代理服务器无需访问远程 Internet,而是通过本地缓存为用户提供服务,有效降低了远程服务器的访问负载,节约了从远程服务器到代理服务器间的网络资源消耗,并能有效降低用户的启动延迟,提高用户接收到的媒体质量。

流媒体的典型应用包括点播电视、视频会议、远程教学、数字图书馆等。(钟玉琢)

liushicidai qudongqi

流式磁带驱动器(streaming tape drive) 在磁带上连续写入或读出多个数据块的一种磁带存储设备。又称流式磁带机。广泛用于硬盘驱动器的后备存储及成批数据交换的场合。记录媒体多采用半英寸带宽的盒式磁带,记录格式与传统快启停磁带机兼容,记录制度相同的记录信息可互换读出。它与传统快启停磁带机在结构上的主要差别是:不使用主动轮及磁带缓冲箱(或缓冲杆),磁带直接用带盘电机驱动,从一带盘传送到另一带盘。由于传动机构简单,在启停磁带时,需要较长的加速、减速时间和磁带运动距离。在标准的记录数据块间隙(IBG)内磁带不做快速启停。

磁带的运动特征是:一个数据块操作结束,IBG通过磁头区时,计算机系统向流式磁带机发出一条随后要执行的命令,使磁带继续运动。此期间称为重置命令时间。如果在重置命令时间内,没有收到随后要执行的命令,磁带进入再定位周期。在再定位周期内,流式磁带机完成磁带的正向减速、反向加速、反向减速以及停带功能,为下一次启动留出足够的磁带长度。

磁带再定位过程如图 1 所示。当磁头读(写)完第 N 个数据块到达 A 点时,磁带继续运动,并准备接收下一条命令。如果到达 B 点前接收到和上次同一类型的命令,磁带不停止运动,继续读(写)第 $N+1$ 个数据块;如果到达 B 点时,还未收到下一条命令

或收到与上一条不同类型的命令,则要进行再定位。磁带从 B 点开始减速到零(C 点),然后反向加速到达 D 点并稳速到 E 点,从 E 点反向减速到 F 点停下,准备下次再启动。从 A 点到 B 点为重置命令时间, B 点到 F 点为磁带再定位时间。

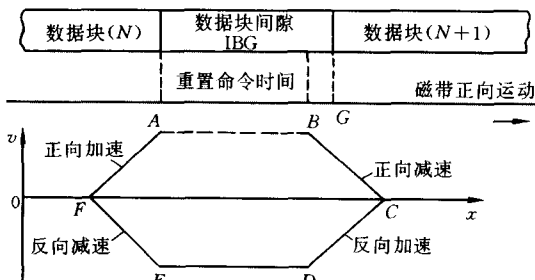


图 1 磁带再定位示意图

此后,若要执行下一个数据块操作,首先从停带点(F)加速磁带,在 G 点前到达正常带速,并开始读出(写入)数据操作。因此,为了在磁带上连续读出(写入)多个数据块,系统必须及时地发出数据操作命令,使磁带保持运动,并避免磁带进入再定位周期。(仇慧珍)

luyouqi

路由器(router) 位于开放系统互连基准(参考)模型(OSI/RM)第三层——网络层的一种网络互连设备(见图 1)。它根据网络层的信息,采用某种路由算法(参见路由选择),为在网络上传送的分组从若干条路由中选择一条到达目的地的路径。

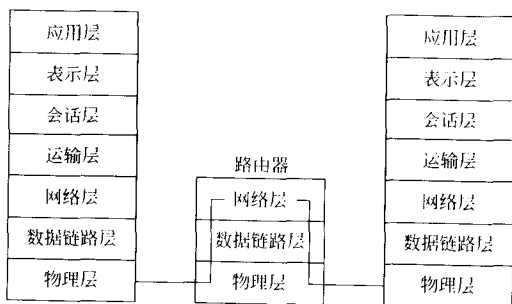


图 1 路由器在网络层上使网络互连

使用路由器可以实现具有相同或不同类型的网络的互连。以前曾把完成这一功能的设备叫做网关,现在把它称作路由器以强调它在 OSI/RM 第三

层上的功能,而网关则用来特指那些在 OSI/RM 的高层(从运输层到应用层)实现不同高层协议之间互相转换的设备。

路由器和网桥的比较 路由器与网桥相比有一系列优点。它具有更强的异种网互连能力、更强的拥塞控制能力、更好的隔离能力等,现分述如下:

(1) 网桥一般只能实现**局域网**之间的互连,路由器不仅可以实现不同类型局域网的互连,而且可以实现局域网和**广域网**,以及广域网与广域网的互连。

(2) 网桥没有流量控制能力,对网络的拥塞的控制能力较差。路由器则具有很强的流量控制能力,可以采用优化的**路由选择**算法来均衡网络负载,从而有效地控制拥塞,避免因拥塞而使网络性能降低。

(3) 网桥只能根据第二层的硬件地址和协议类型来筛选转发的信息,而路由器还可以根据第三层的网络地址、主机地址、地址屏蔽和数据类型来筛选信息,因而具有比网桥更强的隔离能力。这有利于提高网络性能,提高网络的安全保密性,便于根据**网络管理**和维护的需要将一个大的网络分割成若干个独立的部分。

路由器安装复杂,价格昂贵,因此,要根据应用需要和连网环境来选择合适的网桥或路由器。

分类 路由器可以分成单协议路由器和多协议路由器两大类。单协议路由器用于具有相同网络层协议的网络互连;多协议路由器可以支持多种网络层协议,如: IP、IPX、AppleTalk、Banyan VINES 等。

功能 路由器要完成的主要功能是路由选择。路由选择是通过路由器中的路由表来完成的。**路由表**一般可以分为静态和动态两类。静态路由表一般由系统管理员来手工设定,对于路由器的负荷开销比较小,但是不能根据外部网络的变化而自行调整。动态路由表对路由器有一定的开销要求,但是可以根据外部网络的变化而自行调整路由表,不需要系统管理员人为的干涉。

(张世永)

luyou xuanze

路由选择 (routing) 在网络环境中找到一条达到目标计算机的通路的过程。在一个大型网络中,路由选择是十分复杂的,因为一个报文分组可以选择通过许多不同的中间结点到达最终的目标。选择不同的通路(即路由)不但会影响到分组通过网络的传输延迟时间,而且,如果所选择的路由都集中到

网络的某些部分,即某些结点或某些链路,还会造成这些部分的阻塞,影响网络的整体性能或造成分组的丢失。

如何进行路由选择的方法称为**路由选择算法**。路由选择算法有静态路由算法和动态路由算法两大类。静态路由算法又称为固定路由算法。在**静态路由算法**中,所有的通路是在**网络设计**或建设时就预先计算并设定的,在网络运行时不再轻易改变。在大型的复杂网络中,网络的运行状态是动态变化的。比如,不断有新的结点或链路信道加入进来;也会有一些老的结点或链路信道由于故障、维护或其他原因而退出。又如在某个时刻网络中局部地区可能由于突发性的原因而特别拥挤等。在这种网络中通常需采用动态路由算法。它可以依据实时收集的网络运行状态调整路由。最常用的两种**动态路由算法**是**距离向量路由算法**和**链路状态路由算法**。在距离向量路由算法中,每个结点都维护一张以网络中其他所有结点为索引的表。表中每一项包括两个部分:由本结点到达目的结点的可选择的输出线路以及可估算的“距离”。这里,“距离”只是某种度量标准的代名词,具体可以代表跳步数、时间或物理距离等。这张包含“距离”的表就被称为是距离向量。每个结点定期或不定期地与相邻的结点互换距离向量,并依据某种简单的算法来计算出本结点新的距离向量,从而动态地改变路由。这种简单算法的思想是,若某时刻结点 X 知道它到相邻结点 A 的距离是 5,而从 A 收到的距离向量中又告诉它从 A 到目标结点 Y 的距离是 11;同时还知道 X 本身到其相邻结点 B 的距离是 8,而从 B 收到的距离向量中又告诉它从 B 到 Y 的距离是 7,通过计算 $(8 + 7) < (5 + 11)$ 。那么,结点 X 在它的新距离向量中对应以 Y 为目标的那一项中输出线路就选择到 B,且估算其到达 Y 的距离为 $8 + 7 = 15$ 。早期 ARPANET 中使用的 **RIP 协议**就是一种基于距离向量算法的协议,它以跳步数作为“距离”的度量标准。后来,人们发现距离向量路由算法在某些情况下有收敛慢的问题,即要花很长的时间才能真实反映网络中某些变化,从而又提出了链路状态路由算法。在链路状态路由算法中,每个结点测量到它的所有邻点的延迟或开销等链路状态,然后组装成一个链路状态分组发给网络中所有其他的结点。这样,每个结点收到其他结点发送来的链路状态分组后,就具备了网络拓扑结构及全部链路状态的完整信息,并可依此用图论中的最短通路算法,如 Dijkstra 算法,来找出最佳的路

由。链路状态路由算法避免了距离向量路由算法收敛慢的缺点,但计算的开销比较大。国际标准化组织 ISO 采纳的 IS-IS 协议和 Internet 中采用的 OSPF 协议都使用了链路状态路由算法。

通常,大型的计算机网络可以划分为许多自治系统(AS),每个自治系统有自己公共的管理部门和路由策略。这些自治系统有时也称为域。一个大型的网络实际上是许多域的集合。从这个意义上,路由选择协议又可分为内部路由协议和外部路由协议两大类。前述 RIP、IS-IS 和 OSPF 协议都属于内部路由协议,即在域内使用的路由协议。外部路由协议,即在域间使用的路由协议,最著名的外部路由协议是外部网关协议(EGP)和边界网关协议(BGP)。

参考文献

1. Tanenbaum A S. Computer Networks. Third Edition. New Jersey: Prentice Hall, 1996
2. Shay W A 著. 数据通信与网络教程. 高传善等译. 北京: 机械工业出版社, 2000 (高传善)

lüse jisuanji

绿色计算机 (green computer) 为满足环境保护的要求,避免或减少电磁辐射、噪音、化学等污染,保护操作人员的健康,节省能源和原材料消耗,且有助于淘汰报废后回收利用而设计、生产的计算机。

随着计算机在社会中应用日益广泛,计算机对人类社会带来的弊端正日益引起人们的重视。这些弊端有:①环境污染。计算机的高频信号带来电磁辐射,打印机以及其他一些外围设备的噪音,计算机生产和包装使用了有害的化学材料;②电能消耗。全世界目前的计算机已超过 1.5 亿台,耗能很大,尤其是很多办公用计算机电源经常处于开启状态,但并未真正使用,电能浪费很大;③操作人员的职业病。计算机荧光屏对眼睛有害,操作人员长期敲击键盘对手和神经有害。另外,淘汰的计算机很难处置。以上情况都是计算机普及以后给人类社会造成的消极方面的后果。

近年来,全世界环境保护意识日益增强,对计算机产业来说,要求生产出低污染、省电、不影响操作人员健康、有利于陈旧或报废后回收利用的计算机。在这方面,生产厂商已经采取的措施很多,其中共同的主要点有:①采用节电的电源管理技术,在计算机不处于真正工作状态时,自动关闭某些部件的电

源,如使磁盘机、打印机和显示器处于休眠状态。显示器的休眠状态也有利于操作人员眼睛的休整。②电源电压从 5V 降低到 3.3V 甚至更低。③采取各种屏蔽措施,减少高频的电磁辐射。④键盘设计使操作人员的敲击方式避免单调性和适应人体舒适的需求。显示器的频率可以调节以减少显示时的闪烁。⑤在生产计算机的工艺过程中,选用无害的化学材料等。

绿色计算机的发展势头很猛,有些国家已经规定了一些初步指标,凡是符合这些指标的计算机可以附缀“绿色”标记。许多部门已作出规定,优先购买或采用绿色计算机,限制有害的包装材料进入市场。随着环境保护的需要日益迫切,未来的计算机也许绝大多数都是绿色计算机。(李三立)

lunyu lilun

论域理论 (domain theory) 由 D. Scott 于 20 世纪 60 年代创建的指称语义的数学基础。简称域论。

指称语义的特点就是把每个语言成分映射为一个数学对象,然后用此数学对象上的运算来表达该语言成分的语义。但是,这个数学对象往往是递归定义的。例如,考查下列用 BNF 写的表达式定义(参见巴克斯范式):

$EXP ::= NUM | AEXP | CEXP$

$NUM ::= DIGIT | NUM DIGIT$

$AEXP ::= EXP + EXP | EXP - EXP | EXP * EXP$

$CEXP ::= \text{if } COND \text{ then } EXP \text{ else } EXP \text{ fi}$

$COND ::= EXP = EXP | EXP > EXP | EXP < EXP$

$DIGIT ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9$

与之相应的语义域方程是:

$$Exp = Num + [Exp \times Exp \rightarrow Exp] + [Cond \rightarrow Exp]$$

只有解出这个语义域方程,才能得到上述表达式的语义,这里有两个关键问题。第一,是否存在一种数学对象,它是该域方程的一个解? 如果存在,能否把它构造出来? 第二,是否存在一种有效的方法,可以定义这种数学对象上的可计算函数,并且把描述程序设计语言的指称语义所需的函数均包括在内?

D. Scott 对这两个问题都给出了回答。存在着一种叫连续格的数学对象,它可以作为递归域方程的解,存在着一类称作连续函数的函数类,它在连续格上是可计算的。并且这个函数类足够大,可以满足描述一般程序设计语言指称语义的需要。

后来, D. Scott 把连续格上的连续函数理论进一步发展为论域上的连续函数理论。其概要如下。

具有偏序关系 \sqsubset 的集合 D 称为偏序集, 其中 \sqsubset 是 D 上的自反、反对称和传递关系。 D 的子集 M 称为是有向的, 若对每个有限集 $M' \subseteq M$, 存在 M' 的上界 $x \in M$ 。偏序集 D 称为是完备的, 若它的每个有向子集 $M \subseteq D$ 都有一个最小上界 $\sqcup M$, 且 D 有一个最小元素 \perp 。完备的偏序集称为 cpo。 D 的元素 x 称为是紧致的, 如果对 D 的任意有向子集 M , 使得 $x \sqsubset \sqcup M$ 者, 必存在 M 中的元素 y , 使得 $x \sqsubset y$ 。令 $K(D)$ 表示 D 中全体紧致元素的集合, 则 cpo D 称为是代数的, 如果对每个 $x \in D$, 集合 $M = \{x_0 \in K(D) \mid x_0 \sqsubset x\}$ 是有向的, 且 $\sqcup M = x$ 。 D 称为是一个论域, 若 D 是代数的且 $K(D)$ 是可数的。

设 D_1 和 D_2 都是 cpo, $f: D_1 \rightarrow D_2$ 是映 D_1 入 D_2 的函数, 若对 $x \sqsubset y$ 必有 $f(x) \sqsubset f(y)$, 则称 f 是单调的。若 f 是单调的, 且对 D 任意的有向子集 M 有 $f(\sqcup M) = \sqcup f(M)$, 则 f 称为是连续的。如果还有 $f(\perp) = \perp$, 则称 f 是严格连续的, 用 $f: D_1 \rightarrow D_2$ 表示。

由于在指称语义中不仅使用一般函数, 而且使用高阶泛函, 因此上述的论域及连续函数的条件还太一般。例如, 若 D_1, D_2 都是论域, $f: D_1 \rightarrow D_2$ 是连续函数, 则所有这些 f 虽然构成一个 cpo, 但尚不一定构成一个论域。为此需要对论域作进一步的限制。D. Scott 给了一个一般的条件: 可有效表示的论域, 并证明了: 若 D_1 和 D_2 都是可有效表示的论域, 则全体连续泛函 $f: D_1 \rightarrow D_2$ 也构成一个可有效表示的论域。语义域上的操作, 如乘积、求和、 λ 抽象等, 都属于这类连续泛函。利用最小不动点方法即可确定语义域上连续泛函的解。

如果程序中有不确定性, 则程序对同一组数据 (同一个初始状态) 的计算结果将有多种可能性 (不同的后续状态)。用映射来刻画这个程序的执行时, 它的映像就不是由 (通常) 元素构成的集合, 而是由集合构成的集合。这表明, 不能使用通常的平坦论域, 而要使用平坦论域的幂域。为了描述这类程序的指称语义, 需要把上述论域理论推广为幂域理论。

建立幂域理论的关键是把普通论域中的偏序关系推广到幂域中去。在传统的幂集中, $A \subseteq B$ 当且仅当集合 A 包含在集合 B 中。对描述指称语义来说, 仅有这种规定是不够的。两个互不包含的集合之间也可以有偏序关系。对此已引进了三种不同的偏序关系。

1. 霍尔偏序: $A \subseteq_E B$ 当且仅当 $\forall x \in A, \exists y \in B$, 使 $x \sqsubset y$
2. 史密斯偏序: $A \subseteq_M B$ 当且仅当 $\forall y \in B, \exists x \in A$, 使 $x \sqsubset y$
3. 普洛特金偏序: $A \subseteq_{EM} B$ 当且仅当 $A \subseteq_E B$ 且 $A \subseteq_M B$

基于不同的偏序关系, 将得到不同的幂域理论, 它们各有其适用范围。例如, 为了证明程序的部分正确性, 可以使用霍尔偏序; 为了证明程序的健壮正确性 (只要规约正确, 程序一定正确), 可以使用史密斯偏序; 为了证明程序的完全正确性, 可以使用普洛特金偏序。

参考文献

1. 陆汝铃. 计算机语言的形式语义. 北京: 科学出版社, 1992
2. Van Leeuwen J. Handbook of Theoretical Computer Science, Volume B, Formal Models and Semantics. Amsterdam: Elsevier Science Publishers B. V., 1992

(陆汝铃)

luoji biaoshi

逻辑表示 (logic representation) 一种用特定的逻辑公式来表示知识的方式。

符号逻辑公式是形式逻辑语言的合法语句。形式逻辑语言具有严格的语法规则和惟一性的形式语义, 即对合法语句的形式解释。从而可以避免自然语言的歧义性、不明确性等计算机难以处理的问题。逻辑表示是一种比较接近于自然语言的、面向问题的描述性表示方式, 比较容易为人所理解。符号化的形式逻辑系统具有严格、坚实的数学基础。特别是 17 世纪德国数学家、哲学家、逻辑学家 G. W. Leibniz 创导用数学方法研究思维的形式结构及其推理规律而奠定了数理逻辑的基础。此后, 从 18 世纪到 20 世纪 30 年代, 有许多学者开展了系统而深入的研究, 把算术、代数、初等数论、集合论等用于逻辑研究, 创立了如布尔代数、命题逻辑、一阶逻辑等逻辑演算系统。这些逻辑演算系统对于计算机科学技术 and 人工智能的发展, 起了关键性的作用。正因为这些形式逻辑演算系统具有坚实的数学基础, 又比较接近自然语言而容易理解, 所以其中的命题逻辑和一阶逻辑是人工智能最早采用的知识表示方式之一。在 50 年代后期至 60 年代中期, 就已用于研究开发诸如自动定理证明或问题解答等人工智能示范系统。其中较著名的有: 命题逻辑自动定理证明

系统 Logic Theorist; 采用一阶逻辑表示知识, 运用归结原理的通用问题应答系统 QA3 (C. C. Green 等人) 等。命题逻辑和一阶逻辑皆属于形式逻辑, 统称标准逻辑。这类可用于知识表示的符号化形式逻辑系统, 是由一组严格定义的符合语法规则的符号或符号串 (即合法语句) 以及一些严格定义的、对这些合法语句作出解释或进行操作的形式方法 (即形式语义和推理规则) 所组成。其中一阶逻辑是命题逻辑的扩充, 比命题逻辑具有更强的表达能力, 所以更常用于知识表示。

面对极其复杂而千变万化的知识处理问题, 标准逻辑不断暴露出它的种种局限性。于是, 20 世纪中期以来, 针对传统形式逻辑的不足之处, 先后提出了种种新的逻辑体系, 统称非标准逻辑或现代逻辑。例如, 针对传统形式逻辑是精确的、非此即彼 (非真即假) 的二值逻辑, 发展了除真假值外, 还可取未知值或既真又假值等的多值逻辑和按隶属函数在 0 (假)、1 (真) 值之间任意取值的模糊逻辑; 针对传统形式逻辑与情况变化或时间无关的局限, 发展了情境逻辑和时态逻辑; 针对传统形式逻辑的单调性局限, 即其公理系统 (知识库) 的一致性不会因加入新的真命题 (知识) 而遭破坏, 发展了非单调逻辑; 针对传统形式逻辑不能表达事物或其属性的必然性、可能性 (或然性)、应当、也许、允许、禁止、可信、知道等模态命题或模态演算, 发展了模态逻辑。已有的非标准逻辑远不止这些, 新的非标准逻辑还在继续产生。其目的都是为了扩充逻辑的表示能力或提高其推理效能。

和其他知识表示方式相比, 逻辑表示本身, 无论是标准还是非标准逻辑, 还存在若干实质性的问题: ①形式逻辑系统本身存在固有的理论局限。不完备定理早已证明, 包含初等数论的数理逻辑形式系统不可能既是完备的, 又是一致的; Turing 也证明了数理逻辑形式系统本身是不可判定的。尽管为在一定程度上解决或回避这一理论问题, 已经提出了如封闭世界假设、默认推理、限定、广义封闭世界假设等元规则, 但并不能从根本上解决这一理论问题。②形式逻辑系统, 原则上仅适合于进行演绎推理。③符号化形式逻辑系统的精确性、严格性, 使它难以用来表示技巧性、经验性、表象性、常识性知识; ④由于缺乏必要的索引机制而不能直接从知识库检索所需知识, 缺乏处置知识的动态变化和不完备性的有效方法, 面向问题的描述性逻辑表示与面向过程的计算机执行环境不相适应等这样一些问题, 逻辑

表示在传统计算机上的执行效率不高。

总之, 在继续深入理论研究的基础上, 如何扬长避短、增强表达能力、提高执行效能是逻辑表示永恒的研究主题。从应用的角度来看, 针对不同应用领域的知识特点, 如何把逻辑表示与其他知识表示方式, 例如, 面向对象或框架知识表示; 函数表示; 过程表示等更好地结合起来, 力求在增强表达能力的同时, 提高执行效能, 将是重要的发展方向。

参考文献

1. Kowalski R. Logic for Problem Solving. New York: Northholland, 1979
2. 童颖等. 知识工程. 北京: 科学出版社, 1992
(童颖)

luoji chengxu sheji

逻辑程序设计 (logic programming) 编写逻辑程序的方法与过程。逻辑程序由单元逻辑子句 (也称事实, 例如: $p(x)$) 和条件逻辑子句 (也称规则, 如: $p(x) \leftarrow q(x)$, 其中 $p(x)$ 为子句头, $q(x)$ 为子句体) 组成。逻辑程序设计的主要任务在于用单元子句和条件子句描述领域知识中的事实和规则, 作为求解问题的前提, 然后根据推理规则求解问题。

逻辑程序设计源于法国数学家 J. Herbrand 的开创性工作。早在 1930 年, Herbrand 就证明了可以用机械的方法判定逻辑子句的可满足性。1965 年, 美国 J. A. Robinson 提出了面向计算机的合一算法和归结原理。1971 年, R. Kowalski 在上述工作的基础上, 论证了一阶谓词逻辑的子集 (即 Horn 子句逻辑) 可以解释递归过程, 从而开辟了逻辑式程序设计的新领域。遵循 Kowalski 的路线, 法国马赛大学的 A. Colmerauer 等人首先用 **FORTAN** 语言实现了一个自动但效率较低的证明程序, 并命名为 Prolog。70 年代后期, D. H. D. Warren 等人在英国爱丁堡大学实现效率较高的编译型 Prolog 版本。80 年代至今, 逻辑程序设计沿着并行化和约束化两大方向继续发展。

逻辑程序设计的一个重要方面是 Horn 逻辑程序设计。Horn 逻辑程序由 Horn 子句 (即不带否定的谓词公式最多不能超过一个, 例如: $p(x) \leftarrow q(x)$, 其中 $p(x)$ 为子句头, $q(x)$ 为子句体) 组成。Horn 逻辑程序设计的基本内容包括如何构造数据结构, 定义程序和求解问题等。Horn 逻辑程序中数据结构统一为项。项是树形数据结构, 可以表达任意复杂

的数据对象。最简单的项是逻辑常量与逻辑变量。操作数据结构的惟一方法是通过合一操作,合一操作递归地通过项的等同比较,或逻辑变量例化,使得被操作的项变成等同项。它具有变量赋值,结构匹配,结构分解与合成等多种功能。通过定义表达事实的单元子句和表达规则的条件子句定义程序。通过提出问题启动和执行程序。程序的执行过程为 SLD 归结过程。SLD 是 Robinson 一般归结方法在 Horn 逻辑条件下的精化。它是一种反驳式的推理方法。推理序列是逐步递归地构成的,最初结点为原始目标或问题,它是推理序列的起点。以后每一步均由一种计算规则 R 规定从目标中选取一个子目标。使用此子目标和逻辑程序中所有可能的 Horn 子句头合一,实施一步 SL 归结,把产生的归结式作为推理序列的新结点。重复上述过程,直至产生空结点或归结失败为止。

Horn 子句和一般子句的差别在于一般子句允许有否定条件(例如, $p \leftarrow \sim q, r$)。在 Horn 子句逻辑中,不能直接证明一个否定目标。一种特殊的方法是 1978 年 K. L. Clark 提出的有限失败假设,即对于归结失败的目标(如 $(p(x))$),基目标的否定(如 $\sim p(x)$)成立,它是对 SLD 的扩充,称为 SLDNF。逻辑程序设计的一个显著特点是具有非过程性;所有待解问题均是逻辑程序本身的逻辑推论,即任何满足逻辑程序的模型均满足该推论。这就表明,问题求解与程序书写的次序无关,与具体执行过程无关。按照 SLD 推理方法,逻辑程序的解空间有不同的描述方法,例如 OR 树方法,AND 树方法,AND/OR 树方法。它们在刻画和描述程序并行性方面虽有差别。但目的都是为了开发逻辑程序本身存在的各种并行性,主要有子目标的与并行,选择子句的或并行及含公共变量两个子目标之间的流并行,即一个子目标作为生产者,一个子目标作消费者之间的并行。

逻辑程序设计应用:由于逻辑程序设计使用更加面向人类的逻辑语言,以及逻辑程序蕴含的推理机制,使得逻辑程序设计方法获得广泛的应用。逻辑程序设计提供了有关启发式搜索,问题归结和问题求解等简单而有效的模型。逻辑程序设计与语言学的结合产生逻辑文法就是一个典型的例子。逻辑程序设计为专家系统提供单一的简单的知识描述语言,逻辑程序设计语言本身提供了知识库管理机制和使用知识的推理机制,以及专家系统中常见的双向匹配和回溯机制。

参考文献

1. Kowalski R. Logic for Problem Solving. Elsevier North Horlans Inc., 1979
2. Amble T. Logic Programming and Knowledge Engineering. Addison-Wesley Publishers Ltd, 1981
3. Bowen A K. Prolog and Expert Systems. McGraw-Hill Inc., 1991
4. Walker A. Knowledge Systems and Prolog. Addison-Wesley Publishing Company Inc., 1987

(胡运发)

luoji chengxu sheji yuyan

逻辑程序设计语言 (logic programming language) 用于逻辑程序设计的语言。组成逻辑程序的语句的基本形式是 Horn 子句,其形式为

$A \text{ if } B_1 \text{ and } B_2 \text{ and } \dots \text{ and } B_n$

其中 A 是原子公式作为结论,零个或多个原子公式的合取作为条件。若其中任一 $B_i (1 \leq i \leq n)$ 要么是原子公式,要么是原子公式的否定,则称为规范形式。若其中 $B_i (1 \leq i \leq n)$ 可以是任意一阶逻辑公式,则称为一般形式的子句。J. W. Floyd 和 R. W. Topor 已经指明任意一般形式的逻辑子句均可转化为规范形式子句。

逻辑程序设计语言有:①顺序逻辑程序设计语言;②并行逻辑程序设计语言;③约束逻辑程序设计语言。顺序逻辑程序设计语言的代表是 **PROLOG 语言**。Prolog 一个显著的特点是其执行过程有明显的顺序性:子目标 $B_i (1 \leq i \leq n)$ 执行顺序是从左向右,选择适用子句的次序是从上向下,搜索策略是深度优先。在单中央处理器 (CPU) 计算机上,顺序逻辑程序设计语言有较高的执行效率,缺点是求解机制不够完备。

并行逻辑程序设计语言的典型代表有 K. L. Clark 提出的 **PARLOG 语言**。其特点是并行执行所有与目标 $B_i (1 \leq i \leq n)$,并且对于所有满足条件的子句要进行选择提交。推理过程中,一直向前,没有回溯,此语言在多 CPU 机和多机环境下具有较高执行效率和并发通信能力。

约束逻辑程序设计语言 (CLP) 是在顺序或并行逻辑程序设计语言中增加一些特殊原语和推理方法而形成的语言。在逻辑程序执行过程中,这些原语可以自动被延迟或被调用,以便多模式地求解问题。例如求解简单方程的原语,可求解方程 $X = Y + 4$,如

X, Y 未例化, 则该方程被延迟, 一旦 X, Y 中有一个被例化, 则 $X = Y + 4$ 立即被求解。典型的约束逻辑程序设计语言有 PROLOG III, Chip, CLP(R) 等。对某些应用, 比如涉及方程求解的搜索问题时, 通过大量方程的求解, 可删除不必要的搜索空间, 从而大大提高问题的求解速度。

参考文献

Kowalski R A. Predicate Logic as Programming Language. In: Proc. IFIP 74. Amsterdam: North Holland Publishing Co., 1974. 569 ~ 574 (胡运发)

luoji tuiliji

逻辑推理机 (logic inference machine) 可自动进行推理的计算机。它的输入是所要求证明的推理目标、有关的变量以及前提和假设; 输出是关于推理目标的证明结论、有关的解释以及上述变量的值。程序员为逻辑推理机编写逻辑程序, 其中包括推理过程中需要的已知事实, 表示推理规则的逻辑语句和其他形式的语句等。通常, 逻辑程序是用 PROLOG 程序设计语言编写的。逻辑推理机根据上述信息, 按照其本身所具有的推理机制, 选择适当的决策步骤进行逻辑推理演算并输出结果。

逻辑推理机必须是完善的, 即不可能给出错误的推理结果, 又必须是完备的, 即能够给出全部正确推理结果。

逻辑推理机的体系结构可以用 **Warren 抽象机 (WAM)** 语义模型表示。WAM 模型的指令集包含以下 5 类基本指令:

(1) 索引类指令 在一个子句的推理过程中, 用于控制各部分执行次序, 例如, 选择当前子句、回溯、重试等;

(2) 过程类指令 用于管理推理过程中的子句链的选择和环境设置, 以及子句链之间的转移等;

(3) 取检类指令 包括取出参数, 检验子句的形式参数与实在参数是否合一, 记录相应的变量代入关系等;

(4) 设置类指令 在子句体中, 为谓词设置实在参数等;

(5) 合一类指令 用于处理表或函数结构中各分量的设置和合一操作等。

应用这些指令, 可以编写相应的程序来描述逻辑推理机的体系结构; 又可以通过它们, 采用编译、解释以及硬件组成等方法去实现实际的逻辑推理机。

关于逻辑推理机的研究成果有美国的 PLM 机 (1985 年)、日本第五代计算机计划中的 PSI 机 (1985 年) 和 PSI-II 机 (1987 年) 以及具有并行化体系结构的 Multi-PSI 机 (1987 年) 和 PIM 机 (1986 年) 等。

(郑守洪)

luoji yunsuan

逻辑运算 (logic operation) 对有且仅有两种逻辑属性 (“是” 和 “非”, “真” 和 “假”) 的变量以及由这种变量组成的逻辑函数所进行的运算。相应的学科是逻辑代数, 是英国的布尔 (G. Boole) 于 1849 年提出的, 所以也叫布尔代数。

计算机常用的 3 种基本逻辑运算为: “与” (逻辑乘, 符号为 \cdot 或 \wedge)、 “或” (逻辑加, 符号为 $+$ 或 \vee) 和 “非” (求反, 符号为 $\bar{}$, 即在变量的上面加一横杠)。

“与”操作: 当变量 X 和 Y 均为 “1” 时, $X \cdot Y$ 才为 “1”, 否则为 “0”。

“或”操作: 当变量 X 和 Y 任一 (或同时) 为 “1” 时, $X + Y$ 为 “1”, 否则为 “0”。

“非”操作: 当变量 X 为 “1” 时, \bar{X} 为 0; 当 X 为 “0” 时, \bar{X} 为 “1”。

以上 3 种操作的变量可扩大到两个以上, 同时还可以用表达式来替代变量。

从这 3 种基本逻辑操作, 可构造出任意逻辑函数。

在计算机中, 往往设置有专门对两个操作数进行逻辑运算的指令, 例如逻辑加、逻辑乘和异或 (按位加) 等指令, 有时也可能要求其中 1 个数取反后再参与运算。逻辑运算是按位进行的, 即每一位不受操作数中其他位的影响。表 1 是对两个数进行逻辑运算的真值表。

表 1 逻辑运算真值表

A_i	B_i	C_i			
		逻辑加	逻辑乘	异或	同或
0	0	0	0	0	1
0	1	1	0	1	0
1	0	1	0	1	0
1	1	1	1	0	1

表中 A_i, B_i 分别为 A, B 两个操作数中的第 i 位, C_i 为运算结果 C 的第 i 位。

图1(a)是完成3种最基本逻辑运算的逻辑框图(即表示符号)以及相应的逻辑表达式。图1(b)是由3种最基本逻辑电路组成的与非门、或非门、异或门的表示符号以及相应的逻辑表达式。

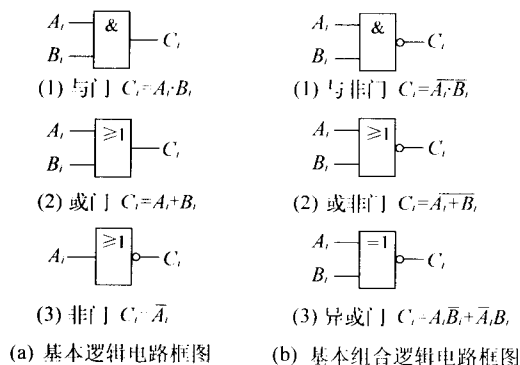


图1 逻辑电路框图

在计算机中央处理器的算术逻辑部件 ALU 中有完成逻辑指令功能的逻辑电路。除此以外,在计算机的各个部件中,还广泛将逻辑电路组合起来完成各种控制功能。

参考文献

王尔乾,巴林凤. 数字逻辑及数字集成电路. 北京: 清华大学出版社, 1994 (王爱英)

luoxuan moxing

螺旋模型 (spiral model) 瀑布模型与演化模型相结合,并加入两者所忽略的风险分析所建立的一种软件开发模型。该模型于 1988 年由美国 TRW 公司 B. 鲍姆(B. W. Boehm)提出。软件项目风险的大小作为指引软件过程的一个重要因素,引入这一概念有可能使得软件开发被看作一种元模型,因为它能包容任何一个开发过程模型。

软件风险是任何软件开发项目中普遍存在的问题,不同项目在此的差别是风险有大有小而已。在制订软件开发计划时,系统分析人员必须回答:项目的需求是什么,需要投入多少资源以及如何安排开发进度等一系列问题。然而若要他们当即给出准确无误的回答是不容易的,甚至几乎是不可能的。但系统分析员又不可能完全回避这一问题。凭借经验的估计给出初步的设想便难免带来一定风险。实践表明,项目规模越大,问题越复杂,资源、成本、进度等因素的不确定性就越大,承担项目所冒的风险也越大。总之,风险是软件开发不可忽视的潜在不利

因素,它可能在不同程度上损害到软件开发过程和软件产品的质量。软件风险驾驭的目标是在造成危害之前,及时对风险进行识别、分析,采取对策,进而消除或减少风险的损害。

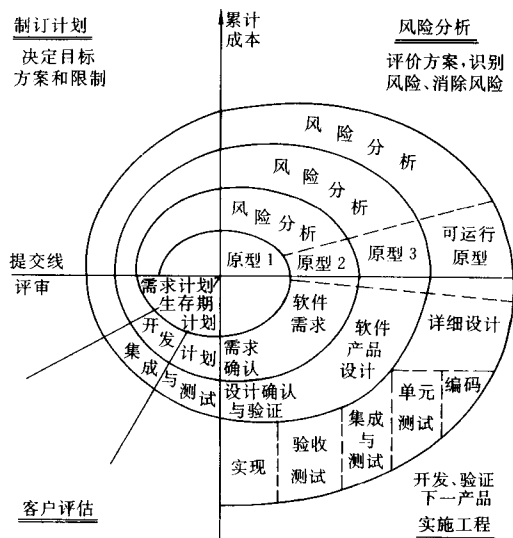


图1 螺旋模型

螺旋模型沿着螺旋线旋转,在笛卡儿坐标的四个象限上分别表达了四个方面的活动(参看图1),即:

- (1) 制订计划——确定软件目标,选定实施方案,弄清项目开发的限制条件;
- (2) 风险分析——分析所选方案,考虑如何识别和消除风险;
- (3) 实施工程——实施软件开发;
- (4) 客户评估——评价开发工作,提出修正建议。

沿螺旋线自内向外每旋转一圈便开发出更为完善的一个新的软件版本。例如,在第一圈,确定了初步的目标、方案和限制条件以后,转入右上象限,对风险进行识别和分析。如果风险分析表明,需求具有不确定性,那么在右下的工程象限内,所建的原型会帮助开发人员和客户,考虑其他开发模型,并把需求作进一步修正。

客户对工程成果作出评价后,给出修正建议。在此基础上需再次计划,并进行风险分析。在每一圈螺旋线上,风险分析的终点作出是否继续下去的判断。假如风险过大,开发者和用户无法承受,项目有可能终止。多数情况下沿螺旋线的活动会继续下去,自内向外逐步延伸,最终得到所期望的系统。图2

给出了螺旋模型的另一图示。

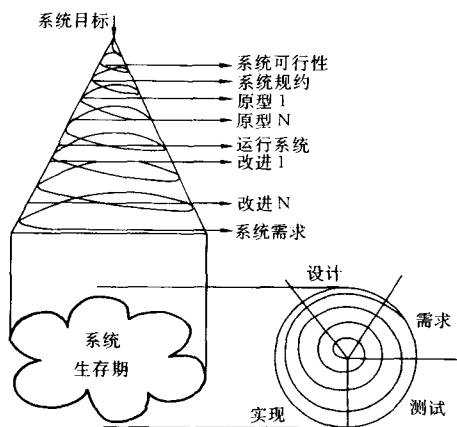


图 2 螺旋模型另一图示

如果对所开发项目的需求已有了较好的理解或较大的把握,无需开发原型,便可采用普通的瀑布模型。这在螺旋模型中可认为是单圈螺旋线。与此相

反,如果对所开发项目的需求理解较差,需要开发原型,甚至需要不止一个原型的帮助,那就要经历多圈螺旋线。在这种情况下,外圈的开发包含了更多的活动。也可能某些开发采用了不同的模型。

螺旋模型适合于大型软件的开发,它是颇为实际的方法,它吸收了 T. Gilb 提出的软件工程“演化”概念。使得开发人员和客户对每个演化层出现的风险均有所了解,并继而作出反应。

和其他模型相比螺旋模型的优越性较为明显,但要求许多客户接受和相信演化方法并不容易。本模型的使用需要具有相当丰富的风险评估经验和专门知识。如果项目风险较大,又未能及时发现,势必造成重大损失。此外,螺旋模型是出现较晚的新模型,远不如瀑布模型普及,要让广大软件人员和用户接受,还有待于更多的实践。

参考文献

- Boehm B. A Spiral Model for Software Development and Enhancement. Computer, 1988, 21 (5): 61 ~ 72
(郑人杰)

M

Mading-Luofu leixing lilun

马丁洛夫类型理论 (Martin-Löf's type theory) 瑞典逻辑学家 P. Martin-Löf 于 1971 年提出的一个类型论形式系统。又称直觉主义类型论或构造类型论。目的在于为 E. Bishop 构造数学提供全面的形式化。

E. Bishop 及其追随者的成果使构造数学得到发展。因此,为这样的构造数学作形式化是极其重要的。这理论的背景思想是基于命题作为类型观点,一命题被解释为一类型,其居元代表该命题的证明。于是谓词逻辑被解释于该类型论中。

P. Martin-Löf 原旨在将类型论构成一个框架,用于解释其他理论,从而类型论的典范化证明直接导出被解释系统的典范化证明。在该理论中,选择公理和一阶海廷算术被巧妙地表示。该类型论有多种表述,各有不同性质和特点,且具有很强的表达能力。例如(居元 a : 类型 A)可分别解释为(元素: 集合)、(证明: 命题)、(程序: 规约)、(解: 问题)。

例如马丁洛夫理论中的规则:

$$\frac{x: A \vdash b: B}{\lambda x: A. b: A \rightarrow B} \quad \frac{m: A \rightarrow B, n: A}{mn: B}$$

按照“命题作为类型,证明作为居元”的观点,可分别表示逻辑规则

$$\frac{A \vdash B}{A \supset B} \quad \frac{A \supset B, A}{B}$$

马丁洛夫类型论是一个规则形式系统,用规则来刻画类型及其行为,有 4 种所谓“判断”: ① A type (A 为类型); ② $a: A$ (a 有类型 A); ③ $a = b \in A$ (类型 A 中居元 a, b 相等); ④ $A = B$ (类型 A, B 相等)。该类型论中定义类型(如自然数类型)以及类型的运算(如 Π, Σ)。在引入类型的同时,类型的典范对象及其计算规则也被引入。用自然推理的式样给出这些规则,且从以下三点论证这些规则: ① 判断形式的语义解释; ② 类型定义; ③ 典范对象的计算规则。对于每个类型,类型运算有规则组(4 条): ① 形成规则(说明可由其他类型形成的类型的形式、记号); ② 引入规则(说明什么是该类型的典范对象); ③ 消去规则(说明如何在该类型上定

义函数,规定引入规则引入的居元才是典范项); ④ 等式规则(说明函数作用于典范对象的计算行为、外延性质)。相等概念有内涵、外延之分,也有内外之分,内相等即通常的“同一”,外相等是定义相等。在引入归约规则后,定义相等是此归约关系的构成和等价闭包。相仿地,也可由判断相等规则来定义。外相等也可以内化,可定义恒等类型(包括内涵的、外延的两类)。内涵恒等类型反映了定义相等,外延恒等类型也力图等同于定义相等,但破坏了内外差别,因此尚有争议。具有内涵定义相等和内涵恒等类型的马丁洛夫类型论是可判定的,具有外延恒等类型的是不可判定的。介乎两者中间的,具有外延定义相等但不具恒等类型的可判定性未知。

近年来,计算机科学家将马丁洛夫类型论作为程序构造的形式理论。在同一个形式系统中能同时表示规约和程序,且由证明规则从规约导出一个正确程序,并给出了验证程序性质的逻辑框架。函数式程序语言如 ML 就符合类型论思想。但是尚未进入程序规约的实用阶段。在构造数学方面,构造集合论的类型解释是很有吸引力的,尤其是否存在这样一个可判定的类型论至今尚未解决。目前已经给出一个反良基公理的马丁洛夫类型论解释,可称是美妙的结果。

20 世纪 90 年代已有不少人涉足马丁洛夫类型理论的语义研究,还有人从事类型理论的扩展以及计算机实现。

参考文献

1. Aczel P. The Type Theoretic Interpretation of Constructive Set Theory; Inductive Definitions. Barcan M R, Dorn G J W, Weingartner P editor. Logic, Methodology, and Philosophy of Science. VII. North-Holland, 1986
2. Martin-Löf P. Intuitionistic Type Theory. Bibliopolis, Naples, 1984

(陆汝占)

meiti chuliqi

媒体处理器 (media processor) 面向多媒体

数据处理的嵌入式可编程微处理器。媒体处理器针对的应用包括音频、视频动态图像、静态图像以及二维和三维图形的实时处理。媒体处理器的具体应用包括各种高清晰数字视频设备、数字电视、DVD 播放机、数字照相机、数字摄像机、机顶盒、视频会议系统、MPEG 编码和解码系统、游戏机、个人多媒体计算机显示卡等。由于多媒体数据处理中以动态视频图像处理的计算量为最大,多数媒体处理器以实时图像压缩和解压缩为优化对象。

发展简史

多媒体处理,特别是数字视频信号处理在 20 世纪 80 年代末期开始广泛地得到推广。最初的数字化的视频信号大多采用通用微处理器进行处理。由于图像处理需要进行大量的数字信号卷积和变换操作,在性能较高的应用中也广泛采用通用的数字信号处理器(DSP)。但因受到当时的通用微处理器和数字信号处理器的运算速度的局限,很难达到实时处理的要求。在此之后,随着 JPEG 和 MPEG 静态和动态图像压缩标准和 H. 263, H. 324 等视频会议标准的出现和消费者对高质量多媒体应用的需求不断增强,媒体处理器在不同的层面上得到了如下几方面的发展:

(1) 面向个人计算机和工作站的微处理器逐步增加了针对多媒体处理的扩充指令集。例如 Intel 的 Pentium-MMX 处理器针对 MPEG 视频解压缩的应用,专门提供了多条专用的指令,从而使得 MPEG 回放所需的指令数量减少。这样,可在个人计算机上实现用软件回放 MPEG 电影节目。HP 公司的 PA-RISC 处理器也采用了类似的指令集扩充方法来提高处理多媒体的性能。从那以后,随着通用微处理器时钟频率的不断提高,个人计算机处理图像、音频、视频信号的性能得到了不断的提高。个人计算机的媒体处理性能的提高也促进了 MP3 等新的多媒体应用的出现。因此,在某种意义上,面向个人计算机的通用微处理器已成为应用最广泛的媒体处理器。但受到处理性能的局限,通用微处理器目前只用于有限的一些媒体处理的应用(如 MPEG 解压缩等),但对计算量大的应用(如 MPEG 压缩)还不能达到实时处理的性能。

(2) 专用的媒体处理电路随着 MPEG 等多媒体技术的标准化而成为可能。这些电路通常只能满足特定的应用需求(如 MPEG 编码器、MPEG 解码器等)。针对具有特定功能的专业视频设备,或具有广大市场的消费产品:如 VCD 播放机、DVD 播放

机、视频电话终端等,专用媒体处理器通过硬件的方法把特定的多媒体处理功能实现在一片或多片超大规模集成电路上。此类媒体处理器具有性能高、价格便宜、省电、系统设计简单等优点。其缺点是功能固定,不能用于其他媒体处理。

(3) 可编程媒体处理器结合了通用微处理器和专用媒体处理器的优点,既可达到很高的处理性能,又有可编程和可适应不同媒体处理要求的特点。通常这一类媒体处理器采用类似于可编程数字信号处理器的高度并行体系结构。处理器内部的不同专用和通用的处理部件可由机器的指令集控制,因而可以通过采用编程的方式来实现特定应用所需的多样的媒体处理算法。由于此类处理器可高速地执行媒体处理中大量的基本操作和具有高度的数据并行性,因而可达到普通处理器所不能及的处理性能。

硬件结构

常见的可编程媒体处理器体系结构可分为两类:数字信号处理器(DSP)体系结构和超长指令字(VLIW)体系结构。

采用 DSP 体系结构的产品有 TI 公司的 TMS320C80 多媒体信号处理器以及 Chromatic Research 的 MPACT 媒体处理器。这一类体系结构的特点是采用多个并行信号处理器(DSP)核来提高媒体处理器的运算速度。如前所述,视频和音频处理涉及大量的数字信号处理算法(如 DCT 数字余弦变换、数字卷积和滤波等),因此采用数字信号处理器核来进行这些运算是一个很自然的选择。但由于单个数字信号处理器核的计算速度不能达到有些应用的实时处理的要求,因此需要采用多个数字信号处理器核进行并行处理。这类数字信号处理器核和普通数字信号处理器类似,通常具有高速算术逻辑部件(ALU)和专用的高速乘法部件以适合卷积等常用的数字信号处理运算。数字信号处理器核的寄存器堆的设计通常和其运算部件相结合——采用分布式寄存器堆。除了通用寄存器以外,还包括了每个运算器专用的寄存器(如乘法结果寄存器等)。这些专用寄存器通常包含在各个运算或功能部件内以提高数据通道访问的速度,且在指令集编码上也比较简洁。数字信号处理器核的存储系统中常采用程序和数据分离的哈佛结构。在数据空间中,通常有较大的高速片内存储器,便于存放当前操作的数据窗口以及中间结果。有些媒体处理器还有专门的外设总线,用以提高多媒体数据的输入输出速率。在媒

体处理器的多个数字信号处理器核之间通常采用高速内部总线、共享存储器或交叉开关等结构进行互连。并行的数字信号处理器核则采用多指令流多数据流作业的方式来进行媒体数据处理,或者采用单指令流多数据流的方式来实现数据并行作业。通常媒体处理器还包括一个控制微处理器以协调各个部件之间的操作。此外,媒体处理器还包括一些高速外设部件和 DMA 控制器,使得多媒体数据得以高效地输入和输出。

基于超长指令字 (VLIW) 体系结构 (参见 VLIW 处理器) 的媒体处理器包括 Philips Trimedia 媒体处理器, Equator 媒体处理器, 以及 Princeton 大学的 VSP 研究计划。这类体系结构与基于 DSP 的媒体处理有很多共同特点。主要的不同在于基本的处理部件不是采用数字信号处理器核, 而是采用超长指令字的指令级并行处理器, 并结合了向量处理器的特点。由于高效地实现具有指令级并行性的应用程序, 与基于多个 DSP 的媒体处理器相比, VLIW 媒体处理器更便于编程。程序员只需要用高级语言 (例如 C 语言) 编写单一指令流的程序, 优化编译器可自动提取程序中的指令级并行性, 并把它映射到 VLIW 处理器的相关部件上, 这样就避免了并行编程的难度。

发展趋势

媒体处理器发展的基本动力在于人们对涉及图形、图像、声音等高质量的多媒体应用的不断提高的广泛要求, 以及多媒体算法的不断更新和改进。以上提到的三类媒体处理器将在不同层面上相对独立地得到发展: 基于通用微处理器的多媒体个人计算机平台将随着处理器主频的提高和体系结构的改进而得到提高和发展; 专用的媒体处理器将随着高清晰度电视和新的数字电视存储标准的成熟而继续为家电产品提供成熟低价的媒体处理方案; 在多媒体算法, 以及数字电视和数字音乐算法发展的阶段, 可编程媒体处理器将为这些算法提供高性能和方便的实现平台。

(廖恒)

Mengtek aluo fa

蒙特卡罗法 (Monte Carlo method) 以概率和统计的理论与方法为基础的一种数值计算方法。

将所求解的问题同一个概率模型相联系, 用计算机实现统计模拟或抽样, 从而求得问题的近似解, 亦称为统计试验法或统计模拟法。蒙特卡罗方法为双重近似, 一是概率模型模拟近似的数

值计算, 二是用伪随机数模拟真正的随机变量的样本。

蒙特卡罗法在 1945 年左右由 J. von Neumann 和 S. M. Ulam 为研制核武器的需要而首先提出来的, 但远在 1777 年, 有人就提出由投针实验求圆周率的方法, 这应该算是最早的蒙特卡罗法。

积分计算 设 $g(x)$ 是 $[0, 1]$ 上连续函数, 且 $0 \leq g(x) \leq 1$, 计算 $S = \int_0^1 g(x) dx$, S 就是由 $g(x)$ 及 $x = 0$, $x = 1$ 与 x 轴所界的图形 G 的面积。考虑在正方形 $[0, 1; 0, 1]$ 上独立的均匀分布随机变量 (ξ, η) , 则 $P((\xi, \eta) \in G) = \int_0^1 \int_0^{g(x)} dy dx = S$ 。如果向正方形 $[0, 1; 0, 1]$ 内均匀投点 n 次, 点的坐标为 (ξ_i, η_i) , $i = 1, 2, \dots, n$ 。计算 (ξ_i, η_i) 落入 G (亦即满足 $\eta_i \leq g(\xi_i)$) 的个数 k , 由强大数定律, $P\left(\lim_{n \rightarrow \infty} \frac{k}{n} = S\right) = 1$, 于是可用 k/n 作为 S 的近似值。这里所谓向正方形投点, 就是构造二维的独立的均匀分布伪随机数列, 并计算 k 。这里 k 是一个随机变量, 是 n 次伯努利试验 (投点) 中, 成功 (点落入 G) 的次数。由于 $E\left(\frac{k}{n}\right) = S, D\left(\frac{k}{n}\right) = \frac{S(1-S)}{n}$ 。且 $\frac{k - nS}{\sqrt{D(k)}}$ 渐近于 $N(0, 1)$ 分布, 因此

$$P\left(\left|\frac{k}{n} - S\right| < \varepsilon\right) \approx$$

$$\Phi\left(\varepsilon \sqrt{\frac{n}{S(1-S)}}\right) - \Phi\left(-\varepsilon \sqrt{\frac{n}{S(1-S)}}\right)$$

这里 Φ 是 $N(0, 1)$ 分布函数。查表可求得 $\Phi(t_\alpha) - \Phi(-t_\alpha) = \alpha$, 于是得 $n \approx t_\alpha^2 S(1-S) / \varepsilon^2$ 。但实际上并不知道 S 的值, 因此必须首先估计 S 。可用试算或

其他办法得到 S 的估计值 \hat{S} , 代入上式求得 n , 再做投点试验, 得到 S 满足精度的估计。从上述 n 与 ε 的关系看出, 增加 n 对结果的精度影响不大, 但如果能改进抽样方法, 降低方差 $D\left(\frac{k}{n}\right) \left(= \frac{S(1-S)}{n}\right)$ 便可提高精度。

还可采用平均值法计算上述积分。任取 $[0, 1]$ 上均匀分布样本 ξ_i , 则 $g(\xi_i)$, $i = 1, 2, \dots$ 也是相互独立同分布的随机变量, 且 $Eg(\xi_i) = \int_0^1 g(x) dx = S$ 。由强大数定律, 可取

$$S_n = \frac{1}{n} \sum_{i=1}^n g(\xi_i)$$

作为 S 的估计。如 $g(x)$ 在 $[0,1]$ 上平方可积,则

$$\begin{aligned} D(S_n) &= \frac{1}{n} Dg(\xi_i) \\ &= \frac{1}{n} \left[\int_0^1 g^2(x) dx - \left(\int_0^1 g(x) dx \right)^2 \right] \end{aligned}$$

这时有近似公式

$$n \approx \frac{1}{\varepsilon^2} t_\alpha^2 D(g(\xi_i))$$

t_α 为 t 分布的临界值。由于 $Dg(\xi_i) \leq S(1-S)$, 可见为了得到同样的精度, 在相同置信度 α 下, 平均值法比投点法所需的计算次数 n 要小。

上述方法不难推广到任意区间 $[a,b]$ 上, 而要求 $0 \leq g(x) \leq 1$ 是非本质的。对于计算高维数值积分

$$S = \int_D \cdots \int g(x_1, \cdots, x_m) dx_1 \cdots dx_m$$

其中 D 为 m 维单位区间, 也可用类似的方法。比如考虑 m 维单位区间上的均匀分布随机向量 $\xi = (\xi_1, \cdots, \xi_m)$, 任取 n 个样本 $\{\xi^i, i = 1, 2, \cdots, n\}$, 则

可用 $S_n = \frac{1}{n} \sum_{i=1}^n g(\xi^i)$ 来近似 S 。

对于低维积分, 一般数值求积法的精度比较高, 但对于四维以上的积分, 蒙特卡罗法就更具优势。在实际应用中, 可以将这两种方法结合起来, 尽可能用分析方法算出对某些变元的积分, 以降低其维数, 然后再用统计试验法求出剩下变元的积分。

蒙特卡罗法还可用于解线性方程组, 矩阵求逆, 常微分方程边值问题求解, 偏微分方程求解, 非齐次积分方程求解, 特征值计算和最优化计算等。

参考文献

1. Shreider Yu A. Method of Statistical Testing (Monte Carlo Method). Amsterdam / London / New York: Elsevier Publishing Company, 1964
2. 中山大学数力系. 概率论及数理统计(下册). 北京: 人民教育出版社, 1980 (金治明)

mimaxue

密码学(cryptology) 以研究数据保密为目的, 对存储或传输的信息进行秘密的变换以防止被第三者窃取的技术。被变换的信息称为**明文**, 它可以是一段有意义的文字或者数据; 变换过后的形式称为**密文**, 密文应该是一串杂乱排列的数据, 从字面上没有任何含义。从明文到密文的变换过程称为加密。变换本身是一个以加密密钥 k 为参数的函数, 记作 $E_k(P)$ 。密文经过通信信道的传输到达目的地后需要还原成有意义的明文才能被通信接收方理解。将密文 C 还原为明文 P 的变换过程称为解密或者脱密。该变换是以解密密钥 k' 为参数的函数, 记作 $D_{k'}(C)$ 。密码学的加密解密模型如图 1 所示。

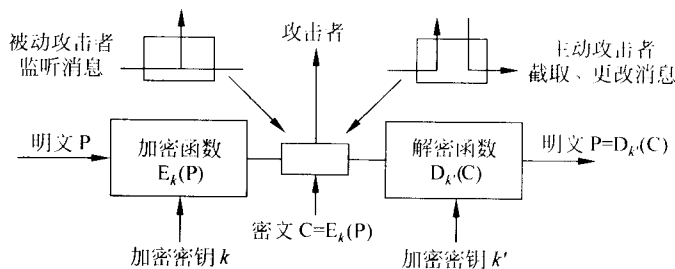


图 1 密码学模型

在传统密码体制中, 加密和解密采用的是同一密钥, 即 $k = k'$, 并且 $D_{k'}(E_k(P)) = P$, 称为**对称密钥密码系统**, 也称**私钥密码系统**。私钥密码系统中使用的密钥常称作**传统密钥**或**传统密码**。现代密码体制中加密和解密采用不同的密钥, 称为**非对称密钥密码系统**, 每个通信方均需要 k 和 k' 两个密钥。在进行保密通信时通常将加密密钥 k 公开, 称为**公钥**, 而将解密密钥 k' 保密, 称为**私钥**, 所以也称为**公钥密码系统**。传统密码系统中最常见的算法有数据加密标准 (DES), 国际数据加密算法 (IDEA) 等。

DES 算法是 IBM 开发的并于 1977 年被美国政府采纳为非机密信息的加密标准。它的原始形式已经在 1995 年被人攻破, 但其修改后的形式仍然是有效的。IDEA 是 Lai 和 Massey 提出的, 目前还没有发现有效的攻击方法。

密码学研究的内容包含两部分: 一是加密算法的设计和研发, 另外一部分是密码分析, 也就是密码破译技术。在密码学中, 如果进行密码分析的攻击者对密码通信进行攻击时, 仅能够被动地监听通信信道上所有信息, 就被称为被动攻击; 如果攻

击者还能够对通信信道上传输的消息进行截取、修改甚至主动发送信息,则称之为主动攻击。攻击者与报文接收方的区别在于是否知道解密密钥。由于攻击者不知道解密密钥,因此很难将密文脱密还原为明文。

公钥方案较对称密钥方案处理速度慢,因此,通常把公钥与对称密钥技术结合起来实现最佳性能。即用公钥密码技术在通信双方之间传送对称密钥,而用私钥密码技术来对实际传输的数据加密、解密。另外,公钥加密也用来对私钥进行加密。

因为在现代密码学研究中对于加密和解密算法一般都是公开的,所以对于攻击者来说只要知道解密密钥就能够破译密文。因此密钥设计被称为密码学研究的核心,密钥保护被视为防御攻击的重点。对于密钥分析来说,对其进行穷举猜测攻击是任何密码系统都无法避免的,但是当密钥长度足够大并且足够随机就会使得穷举猜测在实际上变得不可能。例如,密钥长度为 256 位的加密算法,密钥空间为 2^{256} ,对应为 1 077 量级。如果一台计算机每秒可以对密钥空间进行一亿次搜索,那么全部搜索一遍事件所需的时间以年为单位将大于 10^{62} 。如果密钥空间小或者分布具有一定可预见性,那么攻击者就可能利用相关知识大大缩小搜索空间,从而破译密文。

参考文献

1. 胡道元. 计算机网络(高级). 北京: 清华大学出版社, 1999
2. 卢开澄. 计算机密码学. 北京: 清华大学出版社, 1998 (胡道元)

mianxiang duixiang chengxu sheji

面向对象程序设计 (object-oriented programming) 设计与构造面向对象程序的方法与过程。面向对象程序设计以**对象**为核心,对象是程序运行时刻的基本成分。面向对象程序设计语言中提供了**类**、**继承**等设施。面向对象程序设计即为设计类及由类构造程序的方法与过程,用计算机对象模拟现实世界对象。

现实世界中的系统由一组相互联系、协同作用的基本构件组成,这些基本构件称作客观对象。当用面向对象的方法进行计算机模拟时,在计算机系统中也应有其对应成分,即计算机对象,简称对象。面向对象程序设计方法就是面向这样的对象构造程序乃至系统。

结构化程序设计侧重于功能抽象,它将解决问

题的过程视为一个处理过程,每个模块都是一个处理单位。面向对象程序设计综合了**功能抽象**和**数据抽象**,它将解决问题的过程视为一个分类演绎过程,对象是数据和操作的封装。在结构化程序设计中,过程为一独立实体,显式地为使用者所见;而在面向对象程序设计中,操作是对象私有的,只能通过消息传递来引用。如果参数相同,则过程调用的结果相同,而同一消息的多次发送可能产生不同的结果,取决于对象的当前状态。

面向对象程序设计中的基本概念有对象、类、继承、多态和动态绑定。

参考文献

1. 徐家福等. 对象式程序设计语言. 南京: 南京大学出版社, 1992
2. Shriver B, Wegner P. Research Directions in Object-Oriented Programming. MIT Press, 1987 (张家重 王志坚)

mianxiang duixiang fangfa

面向对象方法 (object-oriented method)

一种把面向对象的思想运用于软件开发过程中,指导开发活动的系统方法。简称 OO 方法,是建立在“对象”概念(对象、类和继承)基础上的方法学。**对象**是由数据和容许的操作组成的封装体,与客观实体有直接的对应关系。一个对象类定义了具有相似性质(属性)的一组对象。而**继承性**是对具有层次关系的类的属性和操作进行共享的一种方式。所谓**面向对象**就是基于对象概念,以对象为中心,以类和继承为构造机制,来认识、理解、刻画客观世界和设计、构建相应的软件系统。

面向对象的方法起源于面向对象的编程语言。20 世纪 60 年代后期 Simula-67 语言中出现了类和对象的概念,类作为语言机制用来封装数据和相关操作。70 年代前期, A. Kay 在 Xerox 公司设计出了 Smalltalk 语言,奠定了面向对象程序设计的基础,1980 年 Xerox 公司推出了商品化的 Smalltalk-80,标志着面向对象的程序设计已进入实用阶段。进入 80 年代相继出现了一系列面向对象的编程语言。如: C++, Object-C, Clos, Eiffel 等。自 80 年代中期到 90 年代, OO 的研究重点已经从语言转移到设计方法学方面,尽管还不成熟,但已陆续提出了一些面向对象的开发方法和设计技术。其中具有代表性的工作有: B. Henderson-Sellers 和 J. M. Edwards 提出的面向对象软件生存周期的“喷泉”模型及面向对象

系统开发的七点框架方法;G. Booch 提出的面向对象开发方法学;P. Coad 和 E. Yourdon 提出的面向对象分析(OOA)和面向对象设计(OOD),J. Rumbaugh 等提出的 OMT 方法学等等。这些方法的提出,标志着面向对象方法逐步发展成为一类完整的方法学和系统化的技术体系。而有关抽象数据类型的基础研究为面向对象开发方法提供了初步的理论。

面向对象方法的具体实施步骤如下:

面向对象分析 从问题陈述入手,分析和构造所关心的现实世界问题域的模型,并用相应的符号系统表示。模型必须简洁、明确地抽象目标系统必须做的事,而不是如何做。分析步骤为:①确定问题域。包括:定义论域,选择论域,根据需要细化和增加论域。②区分类和对象。包括定义对象,定义类、命名。③区分整体对象及其组成部分,确定类的关系及结构。包括:一般-具体结构,整体-部分结构,多重结构。④定义属性。包括确定属性,安排属性。确定实例联结。⑤定义服务。包括确定对象状态,确定所需服务,确定消息联结。⑥确定附加的系统约束。

面向对象设计 因为 OO 方法设计的软件系统结构本质上是并行的,并且突出相对稳定的数据结构,因此 OO 方法与传统的以功能分解为主的方法学的设计内容和步骤有所不同。具体设计步骤如下:①应用面向对象分析对用其他方法得到的系统分析的结果进行改进和完善。②设计交互过程和用户接口。包括:描述用户及任务,并根据需要分成子系统;把交互作用设计成类;设计命令层次;设计交互作用过程及接口,并用相应符号系统表示。③设计任务管理。根据前一步骤确定是否需要多重任务;确定并发性;确定以何种方式驱动任务;设计子系统及任务之间的协调与通信方式;确定优先级。④设计全局资源协调,确定边界条件,确定任务或子系统的软、硬件分配。⑤设计各个类的存储,数据格式;设计实现类所需的算法;将属性和服务加入到各个类的存储对象中;设计对象库或数据库,前四个步骤叫系统设计,最后一步叫对象设计。

面向对象实现 设计阶段设计的对象和关联最终都必须用具体的编程语言或数据库实现。使用 OO 语言来实现 OO 设计相对来说比较容易,因为语言的构造与设计的构造是相似的,OO 语言支持对象、运行多态性和继承。使用非 OO 语言需要特别注意和规定保留程序的 OO 的结构,OO 概念可以映射到非 OO 语言结构中,这只是一个表达方式的问题,

不是语言的能力问题,因为编程语言最终要转换为机器语言,但 OO 语言良好的风格尤为突出。

在传统的面向功能的方法学中,强调的是确定和分解系统功能,这种作法虽然是目标的最直接的实现方式,但由于功能是软件系统中最不稳定、最容易变化的方面,因而获得的程序往往难于维护和扩充,OO 方法首先强调认识来自应用域的对象,然后围绕对象设置属性和操作。用 OO 方法开发的软件,其结构源于客观世界稳定的对象结构,与传统软件相比,软件本身的内部结构发生了质的变化,易用性和易扩充性都得到提高。围绕对象来组织软件和进行软件设计,可将现实世界模型直接自然地映射到软件结构中,可望从根本上解决软件的复杂性问题。并且基于这种新的软件结构,可使软件通过构造的方法自动生成,从而提高软件的生产率和质量。由于软件是建立在应用域自身基础的基本构架之上,不是单个问题的指定功能需求,因而能更好地支持软件需求的演化。

总之,面向对象的开发方法不仅为人们提供了较好的开发风范,而且在提高软件的生产率,可靠性、易用性、易维护性等方面有明显的效果,已成为当代计算机界最为关注的一种开发方法。

参考文献

1. Coleman D. Object-Oriented Development. The Fusion Method. New York: Prentice Hall, 1994
2. Rumbaugh J. Object-Oriented Modeling and Design. New York: Prentice Hall, 1991

(郝克刚 鱼滨)

mianxiang duixiang shujuku

面向对象数据库(object oriented database)

支持面向对象数据库模型的数据库。

面向对象数据库应具备下面的概念与功能:对象、对象标识、类、方法、封装、继承、复合、重载和迟联编、扩展性、计算完备性、持久性、存储管理、并发控制、故障恢复、模式演化、查询。

其中前面 9 个给出了面向对象数据库模型的基本概念,而后面 7 个则给出了传统数据库的基本功能。

面向对象数据库中的数据模型按下面方式组织:

(1) 现实世界中任何事物都可以被统一地模型化为对象,每个对象有一个与其关联的统一的标识叫对象标识。

(2) 每个对象是其状态与行为的封装,其中状态是对象属性值集合,而行为则是在对象状态上操作的方法集合。

(3) 具有相同属性与方法的对象集合构成了类,而类内对象称为实例。

(4) 类属性定义域可以是类,它们构成了类的复合,类具有继承性,一个类可以继承另一个类的属性与方法,该类称为另一个类的子类,而被继承的类称为超类。类的复合与继承构成了一个有向非环结构称为类层次。

(5) 对象是被封装的,它的状态与行为在对象外部是不可见的,外部只能通过用显式定义的消息传递,对其进行操作。

与传统数据库一样,面向对象数据库以消息形式作为数据操纵,它们包括数据的查询、增加、删除、修改等。面向对象数据库也处理并发控制、故障恢复、存储管理等功能。

面向对象数据库不但能支持传统数据库应用也能支持非传统领域的应用,包括 CAD /CAM, OA, CIMS, GIS 以及图形、图像等多媒体领域,工程领域和数据集成等领域。该数据库预计将会在非传统数据库应用中发挥主导作用。

面向对象数据库的产品很多,近期较著名的有 Object Store, O₂, ONTOS 等,其中 Object Store 是面向对象程序设计语言 C++ 的一个持久性扩充,它主要流行于美国,ONTOS 与 Object Store 类似。

O₂ 是一个不依附于其他软件工具的独立的面向对象数据库,它有一个类 SQL 语言,它主要流行于西欧各国。

面向对象数据库的标准化程度不够,目前正在创造条件逐步制订,已有的标准有:ANSI 1991 年的“OODBTG Final Report”,它给出了面向对象数据库管理的参考模型和实现的标准化建议。有关面向对象数据库的终端查询语言标准至今尚未统一,可供参考的标准有:ANSI 的 SQL-3 OMC 的 ODMG。

参考文献

1. Kim W. Introduction to Object Oriented Database System. MIT Press, 1990
2. Kemper A. Object-Oriented Database Management. Prentic Hall Inc., 1994 (徐洁磐)

mianxiang duixiang yuyan

面向对象语言 (object oriented language)

用于描述面向对象程序的程序设计语言。面向对象

程序设计以对象为核心,对象是程序运行时刻的基本成分。语言中提供了类、继承等设施。

面向对象语言借鉴了 20 世纪 50 年代的人工智能语言 LISP,引入了动态绑定和交互式开发环境的思想;始于 60 年代的离散事件模拟语言 SIMULA 67,引入了类的概念和继承;成形于 70 年代的 Smalltalk。面向对象语言的发展有两大方向:一是纯面向对象语言,如 Smalltalk, EIFFEL 等;另一是混合型面向对象语言,即在过程式语言或其他语言中加入类、继承等成分,如 C++, Objec-C 等。

面向对象语言刻画客观系统较为自然,便于软件扩充与复用。其主要特点有四:①识认性,系统中的基本构件可识认为一组可识别的离散对象;②类别性,系统具有相同数据结构与行为的所有对象可组成一类;③多态性,对象具有惟一的静态类型和多个可能的动态类型;④继承性,在基于层次关系的不同类中共享数据和操作。其中,前三者为基础,继承是特色,四者(有时再加上动态绑定)结合使用,体现出面向对象语言的表达能力。

一般认为,较典型的面向对象语言有:①SIMULA 67,支持单继承和一定含义的多态和部分动态绑定;②Smalltalk 支持单继承、多态和动态绑定;③EIFFEL,支持多继承、多态和动态绑定;④C++,支持多继承、多态和部分动态绑定。四种语言涉及概念的含义虽基本相同,但所用术语有别。

参考文献

1. 徐家福等. 对象式程序设计语言. 南京: 南京大学出版社, 1992
2. Special Issue on Object-Oriented Design. CACM. 1990, 33(9) (张家重 王志坚)

mianxiang shuju jiegou fangfa

面向数据结构方法 (data structure-oriented method)

一类侧重从数据结构方面去分析和表达软件需求,进行软件设计的开发方法。该方法从数据结构入手,分析信息结构,并用数据结构图(特指该类方法所用的图形描述工具,例如 Jackson 结构图,Warnier 图)来表示,再在此基础上进行需求分析,进而导出软件的结构。

应用领域的信息域一般包括信息流、信息内容和信息结构等部分。在软件需求分析过程中,根据对信息域分析的侧重点不同就形成了不同的开发方法。面向数据流的开发方法以分析信息流为主,用数据流图来表示信息流;而面向数据结构的开发方

法则以分析信息结构为主,用数据结构图来表示信息结构,并以此为基础进行需求分析,进而导出软件的结构。

面向数据结构的开发方法包括分析和设计两个过程,数据结构在整个过程中起着重要作用。由于很多应用领域的信息都有层次分明的信息结构,系统的输入数据、内部储存数据及输出数据都存在着层次性,并且是相对独立和可区分的,因此,在需求分析过程中可以利用数据结构来分析和表示问题的信息域。在软件设计过程中,不同性质的数据结构往往可以用具有相应的控制结构的程序进行处理。重复性的数据总是用具有循环控制结构的软件来处理,而选择性的数据则由具有条件处理部分的软件来处理,业已证明,仅用三种结构成分即顺序、选择和循环就可以表示所有具有单出口与单入口的程序,因此,可以将具有层次性的数据结构映射到结构化的程序上。

20 世纪 70 年代初,在关于数据结构基础的论述中已出现了面向数据结构的设计思想。

1974 年 J. D. Warnier 发表了《程序的逻辑构造》一文,提出了一种表示数据层次结构的图形工具,即 Warnier 图。他利用顺序、选择、重复三种结构成分来表示数据的层次结构,进而导出程序的结构。1975 年 Michael Jackson 在《程序设计的原则》一文中将输入数据与输出数据的结构在相应层次上对应,并系统地提出了将数据结构映射到程序结构的实用技术。从此,面向数据结构的开发方法便发展起来。

早期的面向数据结构的开发方法主要是针对程序的过程设计而言的,80 年代初逐步发展为较完善的系统化软件开发方法。Warnier 将程序的逻辑构造(LCP)法进一步完善,提出了系统的逻辑构造(LCS)法,用逻辑和形式化方法严格地表示了数据结构,这样便有利于自动生成伪代码,进行系统的校验及优化。同时, Jackson 也将结构化程序设计(JSP)扩充为 Jackson 系统开发(JSD)方法。

1981 年和 1983 年 Ken Orr 对 Warnier 方法进行了扩充,分别发表了《结构需求定义》与《数据结构化程序设计》两文,形成了一种易于理解,并且文档比较完善的系统化开发方法,这就是数据结构化系统开发(DSSD)方法。这一方法涉及到了信息域的所有属性:信息流、信息内容和信息结构。这种方法首先研究应用环境,引入了数据流中的分析方法,以此强化对系统功能特性的分析。同时,也采用了公

式化和综合性的设计方法。

虽然各种面向数据结构的开发方法都有自己独特的表示方式及开发过程,但他们都具有一些共同特点:①这类方法开始于需求分析,并将分析结果作为设计基础,但各种方法并不明显地区分软件的结构设计与过程设计,它们都较早地进入软件的过程表示;②各种方法都为分析人员提供手段来标识关键信息对象(也称为实体或项)和操作(也称为动作或过程);③各种方法都假定信息结构具有层次性;④基本上都采用顺序、选择和重复构造成分表示数据结构;⑤各种方法都提供了一组将层次化的数据结构映射到程序结构的步骤。

面向数据结构的开发方法适用于具有良好定义的,层次分明的信息结构的领域。典型的例子有:商业信息系统开发,其输入和输出有明显的数据结构,并且共同使用同一层次的数据库;操作系统开发,其数据结构由许多有确定结构的表格、文件等构成;CAD/CAM 系统的开发,这方面需要复杂的数据结构用于信息的储存、变换和处理。此外,在工程领域、计算机辅助教育、组合问题求解及其他许多领域都可使用面向数据结构的开发方法。

参考文献

Pressman R S. Software Engineering: A Practitioner's Approach. Third edition. New York: McGraw-Hill Inc., 1992

(郝克刚 葛玮)

mianxiang wenti yuyan

面向问题语言(problem-oriented language)

为了易于描述和求解某类指定问题而专门设计的一种非过程语言。利用它在计算机上解题时,只要指出问题、输入数据和输出格式,就可由相应的计算机系统给出结果。解题过程如图 1 所示。

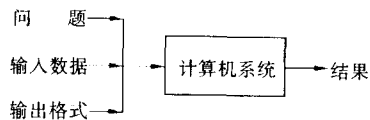


图 1 面向问题语言的解题过程

面向问题语言的同义语有面向应用语言,专用语言,专业化应用语言,或者专业化应用领域的语言。与面向过程语言相比,面向问题语言不要求程序人员描述问题的计算过程或算法。其主要特点是:①申述性。程序人员只要指明“做什么”(及有

关参数),而不必详细说明“如何做”,后者由系统自动解决。②对用户友善。面向问题语言一般是为最终用户设计的,专业性强,用户稍经训练便会使用。③效率高。

早在 20 世纪 60 年代,数值计算的面向问题语言(如线性代数语言)和非数值计算的面向问题语言(如机床控制语言)曾经蓬勃发展。自 60 年代末以来,随着数据库、软件工程、微型计算机和网络等迅猛发展,出现了很多功能强大的面向问题语言,如数据库检索语言,以至诞生了含义更为丰富的名字——**第四代语言 4GL**。

面向问题语言的主要缺点是:①无标准、难移植;②通用性差;③适应性差,不适应计算机资源的扩充升级。

参考文献

1. Truitt T D, Mindlin S B. An Introduction to Nonprocedural Languages: Using NPL. 1983
2. 郭浩志主编. 程序设计语言概论. 长沙:国防科学技术大学出版社,1989 (郭浩志)

minglingshi yuyan

命令式语言(imperative language) 通过指明一系列可执行的运算及运算的次序来描述计算过程的语言。又称强制式语言。

命令式语言以冯·诺依曼式计算机体系结构为背景。**机器语言**与**汇编语言**是最早问世的命令式语言。FORTRAN, ALGOL, COBOL, PASCAL, C, Ada 等高级语言也属命令式语言,其变量对应于存储单元,对变量的访问就是相对应存储单元的访问。各个语句在程序中的顺序以及转向语句等控制语句则明确规定了机器的执行步骤,这就把冯·诺依曼式体系结构的思维方式强加给了程序人员。

高级语言中每种语句几乎都要翻译成若干个低级机器指令。由于存储器空间有限,几乎所有程序都要用重复语句重复执行某些语句,语句的重复执行使命令式语言程序的层次性受到很大的影响。

命令式语言程序的本质是重复地、按步地计算低级(非抽象)值并将之赋给变量(对象),这就迫使程序人员去关心比较低级的细节,而这不适用于设计复杂算法。因此,几十年来命令式语言一直向着隐蔽低级机器属性、提高程序层次与抽象性的方向发展。

命令式语言无论如何发展,都是基于冯·诺依

曼式体系结构的。例如,几乎所有命令式语言都要涉及全程变量、传地址参数(或变量参数、引用参数等),它们一方面可以提高程序的执行效率,另一方面也降低了程序的层次性与抽象性,使程序难以编写、阅读、分析与修改,也使得程序的正确性证明难以进行。随着在非冯·诺依曼式体系结构、基础理论、元器件、人工智能与**软件工程**等方面的不断发展,已有一些非命令式语言(**说明性语言**)问世,函数式语言与逻辑语言就是其中的代表。但几乎没有一个实用的作用式语言是纯作用式的,其中或多或少都要掺入一些具有命令式语言特征的成分。而且,尽管作用式语言有许多优点,但由于它们在效率、应用等方面所存在的诸多问题,命令式语言在相当长的时期内仍会占据主导地位。

参考文献

1. Watt D A. Programming Language Concepts and Paradigms. London: Prentice Hall, 1990
2. 徐宝文. 高级程序设计语言原理. 北京:航空工业出版社,1992 (徐宝文)

mingling yuyan

命令语言(command language) 交互计算机系统向用户提供的一种操作界面的语言。命令语言具有规定的词法、语法和语义,它以命令为基本单位来完成系统提供的各种独立工作任务。完整的命令集所构成的命令语言,反映了该系统向用户提供的功能。

一个命令语言驱动的系统,通常先向用户显示“命令提示符”;随后,用户可输入一条包括参数在内的命令,以实现某任务;在“命令结束符”输入后,系统执行该命令,并给出运行结果或报告出错情况;系统完成该命令后继续显示“命令提示符”,等待用户下一条命令。这种每次执行一条命令的交互式命令语言,与批处理方式的作业控制语言相比,具有明显的优点:简练,灵活,响应速度快,功能易扩充,便于用户根据前一命令结果选择以后的操作。

命令语言已广泛用于各类交互系统,诸如操作系统、正文编辑、数据库操纵、文献资料检索、电子邮件、飞机订票等。目前常用的 UNIX, DOS 操作系统均有命令语言操作界面,shell 作为 UNIX 的统一用户界面是一种典型的命令语言,其命令一般具有以下形式:

\$ 命令名 可选项 文件名 其他参数

其中“\$”为系统的“命令提示符”;可选项是为增加功能而又不增多命令个数的扩展;文件名通常指该命令操纵的对象;命令行的结束符为“换行符”,未标出。shell 命令通常占一正文行,也可占多行(行尾使用“续行符”);一行内可多个命令,只需用“分隔符”分开。shell 还提供许多强的功能:后台作业、输入输出重新定向、shell 变量、命令替换、参数替换、管道线、元字符匹配及可用于编程的多种控制结构(条件、循环)等。

命令语言的设计应从应用的实际情况出发,主要考虑功能需求及使用方便。从“人的因素”观点而论,设计时应考虑以下方面:命令结构一致性,命令名的可读性及缩写策略,提供命令组合、undo 命令、redo 命令、用户自定义命令及创建宏命令的能力。

命令语言方式的弱点是需良好的培训和记忆,有的命令语言过于复杂,有的出错处理功能较差。目前在广泛采用并不断改善命令语言交互方式的同时,还可利用其他交互方式(如选单选项、表格填充、图形方式等),更加方便的自然语言界面正在研究中。

参考文献

1. Shneiderman B. Designing the User Interface. Addison Wesley, 1987
2. 董上海. 计算机用户界面及其工具. 北京: 科学出版社, 1994 (董上海)

mingti luoji

命题逻辑 (propositional logic) 研究由命题与命题联结词构成的更复杂命题,以及这样构成的命题间的推理关系的逻辑。在命题逻辑中,不含命题联结词的命题称为原子命题。不再分析原子命题的内部结构,它们的性质只有真与假的区别。常用 1 和 0 分别代表真命题和假命题,并称集合 {0,1} 为真值集合。因此,可以将命题联结词看作真值集合上的运算。在命题逻辑中经常使用 5 个命题联结词:“非(\neg)”,“与(\wedge)”,“或(\vee)”,“如果…”,“则…(\rightarrow)”和“当且仅当(\leftrightarrow)”。这 5 个命题联结词的运算表如表 1、表 2 所示。

表 1 命题联结词 \neg 的运算表

p	$\neg p$
0	1
1	0

表 2 命题联结词 $\wedge, \vee, \rightarrow, \leftrightarrow$ 的运算表

p	q	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	0	0
1	1	1	1	1	1

由运算表可以看出,这 5 个命题联结词不是互相独立的。可用 \neg 和 \rightarrow 定义 \wedge, \vee 和 \leftrightarrow 。例如, $p \wedge q \equiv \neg(p \rightarrow \neg q)$; $p \vee q \equiv \neg p \rightarrow q$; $p \leftrightarrow q \equiv \neg((p \rightarrow q) \rightarrow \neg(q \rightarrow p))$ 。因此,可取 \neg 和 \rightarrow 为基本联结词。

常用英文字母 p, q, r, \dots 表示命题变元。公式的形成规则如下:

- (1) 每个命题变元是公式;
- (2) 若 A 是公式,则 $\neg A$ 是公式;
- (3) 若 A, B 是公式,则 $(A \rightarrow B)$ 是公式;
- (4) 每个公式都可以经有穷次应用(1),(2)和(3)获得。

只要为公式中的每个命题变元指定了真值,公式就有了确定的真值。如果对于公式 A 中的命题变元赋予任何真值, A 的值均为 1,就称 A 为重言式或永真式,记为 $\models A$ 。例如, $(p \rightarrow p)$ 就是一个重言式。

命题演算是命题逻辑的形式系统,它把重言式组成了一个完全形式化的公理系统。最早的命题演算是 G. Frege, G. Peano 和 B. Russell 于 19 世纪 70 年代至 20 世纪初建立起来的。

在命题演算中,取某些重言式为公理,并规定了某些推理规则,以便从公理出发推出所有的重言式。人们给出了许多不同的命题演算系统,虽然这些系统的公理和推理规则不同,但它们是等价的,即它们推出的定理集都是相同的。下面举出一个这样的命题演算系统。

取以下三种形式的公式为公理:

$$\begin{aligned} & A \rightarrow (B \rightarrow A) \\ & (A \rightarrow B) \rightarrow ((A \rightarrow (B \rightarrow C)) \rightarrow (A \rightarrow C)) \\ & (\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B) \end{aligned}$$

取分离规则为惟一的推理规则,即由前提 A 和 $A \rightarrow B$ 推出结论 B 。命题演算的定理是这样定义的:

- (1) 每个公理都是定理;
- (2) 如果公式 A 和 $A \rightarrow B$ 是定理,则 B 是定理;
- (3) 每个定理都可以通过有穷次应用(1)和(2)获得。若公式 A 是定理,则记为 $\vdash A$ 。

命题演算的公理都是重言式。推理规则保证,

当前提被解释为真命题时,在同样的解释下结论也表示真命题。因此,命题演算的定理都是重言式。这个性质称为命题演算的可靠性,即形式推理可靠地反映了直观的逻辑推理。反之,每个重言式都是命题演算的定理。这个性质称为命题演算的完全性,即形式推理完全反映了直观的逻辑推理而无遗漏。这是 E. L. Post 于 1921 年首先证明的。

参考文献

1. 王宪钧. 数理逻辑引论. 北京: 北京大学出版社, 1982
2. Kleene S C 著. 元数学导论. 莫绍揆译. 北京: 科学出版社, 1984 (何自强)

mohu luoji

模糊逻辑 (fuzzy logic) 研究不确定性(特别是模糊性)推理与知识表示的逻辑基础及其应用的学科。它正处于发展之中,目前主要包括三个方面:狭义模糊逻辑、广义模糊逻辑与模糊(近似)推理。

狭义模糊逻辑是指真值集为单位区间的连续值逻辑或其他一些可能应用于模糊性处理的多值逻辑,它们在语形上与经典逻辑没有本质区别,只是真值集被扩充,基本上不依赖于模糊集理论。对于狭义模糊逻辑的研究,最早可以追溯到 J. Lukasiewicz 20 世纪 20 年代的工作,相继有了较为丰富的内容。特别是 J. B. Rosser 与 A. R. Turquette 在 1952 年就作为未决问题提出了建立超出一阶谓词演算的多值逻辑的设想,并提及多值集的概念,这实际上就是模糊集。到了 1965 年, L. A. Zadeh 在不确定性信息处理的客观背景下,独立地、系统地建立了模糊集理论并指出其在人工智能、自动控制、管理与决策等领域中的应用。正由于此,连续值逻辑及一些相关的领域重新引起了人们的兴趣,一些学者甚至将其改称为模糊逻辑。反过来,狭义模糊逻辑又是模糊集理论乃至模糊数学之逻辑基础;模糊数学的发展是对上述 Rosser-Turquette 问题的部分解答。在模糊集理论出现之后,有关狭义模糊逻辑的工作主要有:①模糊逻辑的归结原理及其在完全分配格上的推广;②模糊逻辑函数的极小化、分析与综合、险态检测及模糊逻辑的函数完备性;③直觉主义模糊逻辑、集合论的形式化系统;④模糊模态逻辑;⑤连续值与剩余格值逻辑的语义方法在不分明拓扑中的应用。

由于狭义模糊逻辑的表达能力有限,不足以处理不确定性推理与知识表示中的许多问题,一些学者致力于发展能刻画语言真值、广义量词、修饰词、

限定词,允许假设与推理规则都具有不确定性,可以作近似证明(推理)的广义模糊逻辑系统,试图为不确定性的处理提供较为完整的逻辑框架。现阶段广义模糊逻辑主要沿下述两个方向独立地发展,但离建立能统一这两个方面工作的完整形式化演算还有相当的距离。

(1) 70 年代以来 L. A. Zadeh 等以模糊集为工具提出了一种对语言真值、模糊量词、修饰词与限定词作语义解释的翻译规则。①语言真值被解释为单位区间内的模糊数,作为其简化,有时也考虑单位区间的子区间。②模糊量词通常利用模糊集的基数刻画,此即比例法。这种方法虽然十分自然,但逻辑性质很不理想。另一种局部化地刻画模糊量词的方法是代换法,它只适用于有限论域。为了克服这些困难,有的学者提出了用 Sugeno 模糊测度与积分刻画模糊量化命题的方法。③模糊修饰词是用来表现语气副词的,它适度地改变命题的意义,通常以平移或指数函数刻画。④限定词有真值、概率与可能性限定三种类型。1985 年刘叙华建立了算子模糊逻辑,其原始目的可以看作处理限定词的一种简化的形式化方法,后在不同的直观意义下产生了多种变形。目前,这方面的工作还是初步的;除了刘叙华已经证明算子模糊逻辑中归结原理的完备性外,对于带有语言真值、模糊量词、修饰词、限定词的广义模糊逻辑的完全形式化还有许多困难的问题有待解决。由于在自然语言理解等方面的潜在应用价值,语言真值、模糊量词、修饰词与限定词的数学处理及其与 Keisler 概率量词等的关系值得深入研究。

(2) 1979 年 J. Pavelka 提出了假设带有不确定性的希尔伯特型完备格值演绎系统,并在有限链与连续值情形获得了有关相应命题逻辑的完备性与不完备性的结果。其后,人们证明了对应的一阶模糊逻辑的完备性;将 Pavelk 的希尔伯特型演绎系统推广到 Gentzen 型模糊自然演绎或推理规则具有不确定性的情形;建立了允许假设与推理规则作近似匹配的数值化近似证明理论,并证明了带有这种近似证明(但真值仍取二值的)命题与一阶逻辑的完备性。这方面仍有许多问题需要讨论,如各种不确定性的组合机制与推理过程中推理规则按重要性程度的排序。这些问题的解决可能为人工智能系统的基本积木块——产生式系统中不确定性处理奠定较为完整的逻辑基础。

将模糊逻辑应用于智能化电器产品的开发与工业过程控制,这些实际应用的理论基础是模糊推理,

它与狭义、广义模糊逻辑之间没有太多的本质联系。1972年, L. A. Zadeh 提出了模糊推理的关系合成规则, 现有的绝大多数模糊推理方法都是关系合成规则的变形(如真值限定方法)或扩充(如多维、多条件及区间值模糊推理)。70年代后期, E. H. Mamdani 等将模糊推理方法应用于模糊控制器的设计, 这为80年代后期模糊技术的兴起打下了基础。近年来, 模糊逻辑与神经网络的结合受到了人们很大的重视, 它反映了人工智能中符号主义与联接主义机制的综合集成; 带模糊量词与修饰词的模糊推理、模糊推理在类比推理中的应用、模糊推理的摄动及其对模糊控制器稳定性的影响也开始引起人们的注意。设 U, V 是两个论域, x, y 分别是取值于 U, V 的变元, A, A' 是 U 的模糊子集且 B 是 V 的模糊子集。模糊推理的基本模式是

Ant. If x is A then y is B
 x is A'

Cons. y is $B' \triangleq C(A, B; A') = ?$

这里结论 $B' = C(A, B; A')$ 是 V 的模糊子集, 它通常由关系合成规则给出如下:

$$C(A, B; A')(y) = \sup_{x \in U} A'(x) T(A(x) \rightarrow B(y)), y \in V$$

其中 T 是合取算子, \rightarrow 是蕴涵算子, 由于结论 $B' = C(A, B; A')$ 依赖于算子 T, \rightarrow 的选取, 在实际应用中如何根据实际问题恰当地选取合取与蕴涵算子往往对于应用效果产生决定性的影响; 许多学者已经对一些常用的合取与蕴涵算子分析比较了模糊推理的结果。在理论上, 因为关系合成方法不满足分离规则 $C(A, B; A) = A$, 人们通常认为这是模糊推理的不合理性。一方面, 一些学者利用模糊关系方程的理论考虑了对于什么合取与蕴涵算子、在什么情况下分离规则成立的问题; 另一方面, 可以证明对于绝大多数常用的合取与蕴涵算子, 每个模糊推理模式等价于另一个满足分离规则的模式, 从而揭示了其隐藏在表面不合理性之后的深刻的合理性。

参考文献

1. Zadeh L. A. Fuzzy Sets and Applications. New York: John Wiley & Sons, 1987
2. Pavelka J. On Fuzzy Logic I, II, III. Zeitschr. f. math. Logik und Grundlagend. Math., 1979, 25(1): 45~52, (2): 119~134, (5): 447~464
3. 刘叙华. 基于归结方法的自动推理. 北京: 科学出版社, 1993 (应明生)

mohu tuili

模糊推理 (fuzzy reasoning) 根据模糊的事实, 通过模糊的推理过程, 获得可接受的模糊结论的一种推理方式。

从1965年 L. A. Zadeh 提出模糊集理论以后, 建立在模糊集上的各种模糊数学得到了迅速发展。尤其是1971年 R. C. T. Lee 和 C. L. Chang 提出了模糊逻辑以后, 建立在模糊逻辑上的模糊推理的研究受到了计算机科学家, 特别是人工智能学者的重视。人在现实世界中经常面对一些模糊的、不完全的信息, 并使用一些模糊的推理规则, 作出可以接受的合理判断。要让计算机模拟人的智力活动, 仅仅让计算机具有传统的数学推理能力是不够的, 它必须能够像人一样, 对不精确信息能够进行“合理”的模糊推理。因此, 模糊推理在计算机科学中的重要性不低于传统的精确推理。

模糊集理论

模糊集 设 U 是论域, U 的一个模糊子集 A 是由如下的隶属函数 μ_A 决定的:

$$\mu_A: U \rightarrow [0, 1]$$

对任意 $u \in U, \mu_A(u)$ 称为 u “属于” U 的等级。

显然, μ_A 是集合特征函数的推广。为方便计, 下面的叙述是在假设 U 为有限集合下进行的。设 $U = \{u_1, \dots, u_n\}$, U 的模糊子集 A 可表示为:

$$A = \frac{\mu_A(u_1)}{u_1} + \dots + \frac{\mu_A(u_n)}{u_n}$$

或者, 当 u_i 不是数字时, A 可更简单地表示为:

$$A = \mu_A(u_1)u_1 + \dots + \mu_A(u_n)u_n$$

其中 $\mu_A = 0$ 的项可省略。

例如, $U = \{a, b, c, d\}$, U 的一个模糊子集 A 如下:

$$A = 0.3a + 0.8c + 0.5d$$

相等 U 的两个模糊子集 A, B 称为相等, 记为 $A = B$, 如果 $\mu_A = \mu_B$ 。称 A 是 B 的子集, 记为 $A \subseteq B$, 如果 $\mu_A \leq \mu_B$ 。例如:

$$U = \{a, b, c, d\}$$

$$A = 0.5a + 0.8b + 0.3d$$

$$B = 0.7a + b + 0.3c + d$$

于是, A 是 B 的子集。

模糊关系 若 $U = U_1 \times U_2$ (集合的笛卡儿积), 则 U 的任一模糊子集都称为 U 中一个二元模糊关系 R , 即

$$R = \sum \frac{\mu_R(u_1, u_2)}{(u_1, u_2)}$$

二元模糊关系也可以用模糊矩阵表示。

例如, 设 $U_1 = U_2 = \{1, 2, 3, 4\}$, “远大于” 关系 R 可表示成: $R = 0.3 / (1, 2) + 0.8 / (1, 3) + 1 / (1, 4) + 0.8 / (2, 4) + 0.3 / (3, 4)$, 也可以表示成如下矩阵:

$$R = \begin{bmatrix} 0 & 0.3 & 0.8 & 1 \\ 0 & 0 & 0 & 0.8 \\ 0 & 0 & 0 & 0.3 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

设 R 是 $U \times V$ 中模糊关系, S 是 $V \times W$ 中模糊关系, R 与 S 的合成关系如下:

$$R \circ S = \sum \frac{\max_{v \in V} \{ \min \{ \mu_R(u, v), \mu_S(v, w) \} \}}{(u, w)}$$

例如

$$\begin{matrix} R & S & R \circ S \\ \begin{bmatrix} 0.3 & 0.8 \\ 0.6 & 0.9 \end{bmatrix} \circ \begin{bmatrix} 0.5 & 0.9 \\ 0.4 & 1 \end{bmatrix} & = & \begin{bmatrix} 0.4 & 0.8 \\ 0.5 & 0.9 \end{bmatrix} \end{matrix}$$

合成运算“ \circ ”实际上就是矩阵乘法运算, 只不过将其中的“乘”换为“ \min ”, “加”换为“ \max ”。

和普通集合一样, 模糊集之间也可以引进余、并、交及一些特殊运算。

余运算 设 A 是模糊集, 它的余集 \bar{A} 定义为如下的模糊集:

$$\bar{A} = \sum \frac{(1 - \mu_A(u))}{u}$$

例如, $U = \{a, b, c, d, e\}$, $A = 0.5a + 0.7c + 0.1d$, 于是, $\bar{A} = 0.5a + 0.3c + 0.9d + b + e$ 。

并运算 设 A, B 是两个模糊集, 它们的并集 $A \cup B$ 定义为如下模糊集:

$$A \cup B = \sum \frac{\max \{ \mu_A(u), \mu_B(u) \}}{u}$$

例如, $U = \{a, b, c, d, e\}$, $A = 0.5a + 0.1c$, $B = 0.3a + 0.2b$, 于是 $A \cup B = 0.5a + 0.2b + 0.1c$ 。下面, 并运算也使用符号“+”, 写成 $A + B$ 。

交运算 设 A, B 是模糊集, 它们的交集 $A \cap B$ 定义为如下的模糊集:

$$A \cap B = \sum \frac{\min \{ \mu_A(u), \mu_B(u) \}}{u}$$

例如, U 如上, $A = 0.5a + 0.1c$, $B = 0.3a + 0.2b$, 于是 $A \cap B = 0.3a$ 。

模糊集中的交、并运算, 有交换律、结合律、分配律和吸收律。

化模糊算子 K 设 A 是模糊集, K 作用在 A 上产生另一个模糊集

$$F(A; K) = \sum \mu_A(u) \cdot K(u)$$

其中 $K(u)$ 是 K 作用在 u 上产生的模糊集, \cdot 是乘法。

例如 $U = \{1, 2, 3, 4\}$

$$A = \frac{0.8}{1} + \frac{0.6}{2}$$

$$K(1) = \frac{1}{1} + \frac{0.4}{2}$$

$$K(2) = \frac{1}{2} + \frac{0.4}{1} + \frac{0.4}{3}$$

于是

$$\begin{aligned} F(A; K) &= 0.8 \times K(1) + 0.6 \times K(2) \\ &= 0.8 \times \left(\frac{1}{1} + \frac{0.4}{2} \right) \\ &\quad + 0.6 \times \left(\frac{1}{2} + \frac{0.4}{1} + \frac{0.4}{3} \right) \\ &= \frac{0.8}{1} + \frac{0.6}{2} + \frac{0.24}{3} \end{aligned}$$

化模糊算子在定义程度词时有用。

带有非模糊真值的模糊逻辑

为方便计, 下面只考虑命题逻辑。设 \mathcal{L} 是经典的命题逻辑, A 是一个原子。若把 A 的真值范围改为区间 $[0, 1]$, 则称此新的逻辑 \mathcal{L} 为一个模糊逻辑。用 $\tau(A)$ 表示 A 的真值。

\mathcal{L} 中公式的真值按如下规则计算:

$$\tau(\sim G) = 1 - \tau(G)$$

$$\tau(G \wedge H) = \min \{ \tau(G), \tau(H) \}$$

$$\tau(G \vee H) = \max \{ \tau(G), \tau(H) \}$$

$$\tau(G \rightarrow H) = \tau(\sim G \vee H)$$

$$\tau(G \leftrightarrow H) = \tau((G \rightarrow H) \wedge (H \rightarrow G))$$

公式的解释: 设 G 是一个公式, 对 G 中每个原子赋予一个真值, 这组赋值称为 G 的一个解释。按照上面给出的规则, 公式 G 在解释下有一个确定的真值。

例如, $G = (A \vee B) \wedge \sim C$, 解释 $I = \begin{pmatrix} A & B & C \\ 0.8, & 0.6, & 0.3 \end{pmatrix}$, 在这个解释下, 公式 G 的真值 $\tau(G) = 0.6$ 。

模糊真(假) 如果公式 G 在某解释 I 下, 有 $\tau(G) > 0.5$, 则称 G 是模糊真的; 如果有 $\tau(G) < 0.5$, 则称 G 是模糊假的; 如果有 $\tau(G) = 0.5$, 则 G 称为不定。

在模糊逻辑 \mathcal{L} 中, 可进行模糊三段论推理, 亦即, 若事实 A 是模糊真的, 推理规则 $(A \rightarrow B)$ 也是模糊真的, 则可推出模糊真的结论 B 。

带有模糊真值的模糊逻辑

在中原子的真值有时不能用 $[0,1]$ 区间中的一个数表示,需要用模糊集表示。例如, A 的真值是“比较年轻”,描述“比较年轻”的恰当方法是给出一个模糊集,而不是给出 $[0,1]$ 区间中的一个数。这就是带有模糊真值的模糊逻辑。

在上面给出的模糊三段论推理中,要求知道 A 和 $(A \rightarrow B)$ 都是模糊真的。但是,在模糊推理中有时会遇到这种情况:已知 A 和 $(A^* \rightarrow B)$ 是模糊真的, A^* 是 A 的一个近似,那么我们能否推出 B 是模糊真的?我们做如下处理:将规则 $(A^* \rightarrow B)$ 看作是一个二元模糊关系, A 是一元模糊关系,用这两个模糊关系的合成 $A \circ (A^* \rightarrow B)$ 作为 A 和 $(A^* \rightarrow B)$ 所推出的结论 B 的模糊描述。

例如,设 $U = V = \{1, 2, 3, 4\}$

$$A = "u \text{ 是小的}" = \frac{1}{1} + \frac{0.6}{2} + \frac{0.2}{3}$$

$F = "u \text{ 和 } v \text{ 近似相等}"$

$$\begin{aligned} &= \frac{1}{(1,1)} + \frac{1}{(2,2)} + \frac{1}{(3,3)} + \frac{1}{(4,4)} \\ &\quad + \frac{0.5}{(1,2)} + \frac{0.5}{(2,1)} + \frac{0.5}{(2,3)} + \frac{0.5}{(3,2)} \\ &\quad + \frac{0.5}{(3,4)} + \frac{0.5}{(4,3)} \end{aligned}$$

于是

$$B = A \circ F$$

$$\begin{aligned} &= (1 \quad 0.6 \quad 0.2 \quad 0) \circ \begin{pmatrix} 1 & 0.5 & 0 & 0 \\ 0.5 & 1 & 0.5 & 0 \\ 0 & 0.5 & 1 & 0.5 \\ 0 & 0 & 0.5 & 1 \end{pmatrix} \\ &= (1 \quad 0.6 \quad 0.5 \quad 0.2) \end{aligned}$$

设化模糊算子 $K = "有点"$,并且

$$K(1) = \frac{1}{1} + \frac{0.7}{2}$$

$$K(2) = \frac{1}{2} + \frac{0.7}{3}$$

$$K(3) = \frac{1}{3} + \frac{0.7}{4}$$

$$K(4) = \frac{1}{4}$$

于是,“有点小” $= F(\text{小的}; K) = (1 \quad 0.7 \quad 0.42 \quad 0.14)$,此结果近似于 B 。所以, B 可以认为是“有点小”。

因此,使用模糊关系的合成作为模糊推理规则,我们从“ u 是小的”和“ u 和 v 近似相等”推出了“ v 有

点小”的结论,这很符合直观。

可以看出,模糊逻辑的研究还刚刚开始,所得结果也还是初步的。将命题,公式的模糊性局限在它们的真值上,没有充分显示出模糊逻辑和经典逻辑的区别,进一步需要研究的问题还很多。但是它在计算机科学中的重要性已经是很明显的了。

参考文献

1. 刘叙华. 模糊逻辑与模糊推理. 长春: 吉林大学出版社, 1989
2. Zadeh L. A 著. 模糊集合、语言变量及模糊逻辑. 陈国权译. 北京: 科学出版社, 1982 (刘叙华)

mokuaihua fangfa

模块化方法 (modular method) 一种软件开发方法,把一个待开发的软件分解成若干小的简单的部分,称为模块。每一个模块都独立地开发、测试,最后再组装出整个软件。这种开发方法是对待复杂事物的“分而治之”的一般原则在软件开发领域的具体体现。

模块化开发方法涉及的主要问题是:模块设计的规则,系统如何分解成模块。

模块是执行一个特殊任务或实现一个特殊的抽象数据类型的一组例程和数据结构。模块通常由两部分组成。**接口**:列出可由其他模块或例程访问的常数、数据类型、变量、函数等;实现:私有量(只能由本模块自己使用的)及实际实现本模块的源程序代码。

模块的接口部分刻画了各个模块是如何耦合的,是其他模块的设计者和使用者所需要知道的。而实现部分是各模块的内部事务,其他模块并不需要知道。这也体现了对待复杂事物的另一原则——抽象原则(在软件领域称为信息隐蔽原则),即把非本质的性质隐藏起来,只突出那些本质的性质,以减轻人们思考和注意的负担。模块化澄清和规范化了软件中各部分间的界面。如此就便利了成组的软件设计人员工作,也促使了更可靠的软件设计实践。

在把系统分解成模块时,应该遵循以下的规则:

- ①得到最高的模块内聚,也就是在一个模块内部的元素最大程度地关联。只实现一种功能的模块是具有最高内聚的,具有三种以上功能的模块则是低内聚的。
- ②最低的耦合,也就是不同模块之间的关系尽可能弱。
- ③模块大小适度。
- ④模块调用链的深度(嵌套层次)不可过多。
- ⑤接口干净,信息隐蔽。
- ⑥尽可能地复用已有模块。

如何对一个规约进行分解,以得到模块化的系统结构。已经有一些基于设计规则的方法。

(1) 数据结构设计方法 最著名的是 Jackson 结构化设计(参见 Jackson 系统开发方法)。它从画出所有输入/输出数据的逻辑结构图开始,最后得到程序结构图,反映了系统结构。可能还需要继续对其中模块求精,得到更低级的模块,但是基本程序结构是不变的。

(2) 功能分解 步骤是:陈述出功能意图(即要解决的问题),进行功能求精(即划分层次),连接求精了的功能,进行检查,再求精,再检查,直至得到满意的解决为止。这种方法,也称为结构化设计、层次化分解、模块分解、功能分解。

(3) 数据流设计 步骤是:把问题分解成由动作图(也称为进程)和数据图(也称为流)组成的数据流图,还有存放待处理的静态信息的存储元素。然后从数据流图中找出中心进程,以它为根,把数据流图转换为树形结构。按照功能分解形式来分解进程,即把模块分为三类(输入、变换、输出),进行进程求精,得到 PDL 语言表示。最后把 PDL 变成某种程序语言。

(4) 面向对象的设计 这一方法要求标识出对

象及其属性、每个对象所需要的操作。把数据及函数封装在一起,以形成类。并建立其间相互可见的关系,即许可的调用与被调用关系,形成每个类的界面。最后是实现每个类(参见面向对象方法)。

模块化方法的优点是明显的,难点所在是如何处理大型问题。分而治之的原则是好的,然而对大型问题会难以找到下手之处。当从一个角度看问题时,要求隐蔽的信息是这些,而换一个角度看时,却要求隐蔽另外的信息。模块化方法在发展的早期主要是手工设计方法,以后发展了许多自动化工具来支持它们。最好是几种方法组合使用,各自发挥所长,但是内部表示和工具间的相容性是一大困难。

参考文献

Lewis T G. CASE: Computer-Aided Software Engineering. New York: Van Nostrand Reinhold, 1991

(董煜美)

mokuai jiegou tu

模块结构图(modular structure diagram)

一种表示软件各模块之间控制关系的图形工具。如图 1 所示。

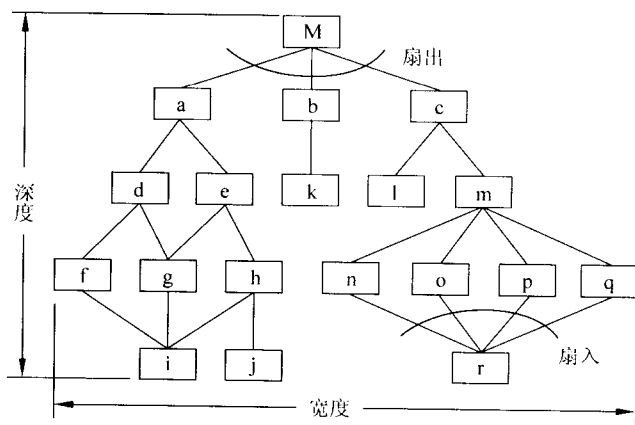


图 1 模块结构图

其中,矩形表示一个软件模块,矩形之间的线段表示模块之间的控制关系。按控制关系,可以把模块分为若干个层次。上一层次的模块称为控制模块,而下一层次的模块称为从属模块。模块结构图的度量属性,主要包括模块结构图的宽度、深度以及模块的扇入和扇出。模块结构图的宽度是指具有最多模块

的那一层之模块数目,或称模块结构图的控制跨度;模块结构图的深度是指模块控制层的数目,或称模块结构图的控制深度;模块的扇入是指其控制模块的数目;模块的扇出是指其从属模块的数目。

在实践中,模块结构图通常用于软件系统概要设计、标识系统的模块结构。(李宣东 王立福)

moni jisuanji

模拟计算机 (analog computer) 可对连续物理变量进行并行数学运算的解算装置。当把模拟计算机的各种不同功能部件按照一个物理系统的数学描述连接时,就构成这个物理系统的数学模型,可用来直接模仿该物理系统的行为。按模拟计算机中代表变量的物理量性质和运算部件构造的不同,模拟计算机可分为机械式、机电式或电子式等不同类型。在模拟计算机中,加、减、乘、除、微分、积分等数学运算,皆由相应的运算部件,如加法器、乘法器、积分器等并行完成。按运算的性质,运算部件可分成线性和非线性两大类。

在电子模拟计算机中,变量为连续变化的直流电压、电流或者电荷。各种运算部件主要是由运算放大器和精密电阻、电容以及特殊的开关元器件等构成。运算放大器是构成各种运算部件的核心器件。它由一个高放大倍数的直流放大器 A 和输入阻抗 Z_i 以及跨接于输出与输入之间的反馈阻抗 Z_f 组成,如图 1 所示。按克希霍夫定律和欧姆定律,可得出其回路方程如下:

$$\frac{E_o - V}{Z_f} + \frac{E_i - V}{Z_i} = I$$

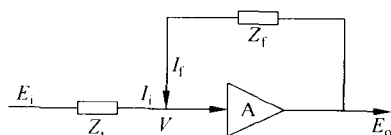


图1 运算放大器框图

式中, E_i , E_o 分别为输入、输出电压, I 为电流。当直流放大器的放大倍数很大 ($> 10^7$), Z_i 及 Z_f 也足够大时, $E_o \gg V$, $E_i \gg V$, $I \approx 0$ 。于是可得出模拟计算的基本关系:

$$\frac{E_o}{E_i} = -\frac{Z_f}{Z_i}; \quad E_o = -\frac{Z_f}{Z_i} E_i$$

若阻抗 Z_i , Z_f 皆为纯电阻, 令为 R_i , R_f , 且 $R_i = R_f$, 便构成一个反相器; 一般情况下, $R_i \neq R_f$, 便可构成一个比例运算器。若 Z_i 是多个电阻的并联 (图 2(a)), 可导出如下关系:

$$E_o = -\left(\frac{R_f}{R_1} E_1 + \frac{R_f}{R_2} E_2 + \frac{R_f}{R_3} E_3 + \cdots + \frac{R_f}{R_n} E_n\right)$$

这就构成一个比例加法器; 若 Z_f 为一个电容器 (图 3(a)), 便可得出以下动态回路方程:

$$V_o = \frac{q}{C} = -\frac{1}{C} \int_0^t I dt = -\frac{1}{RC} \int_0^t V_i dt$$

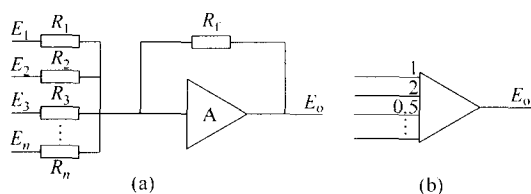


图2 比例加法器框图及符号图

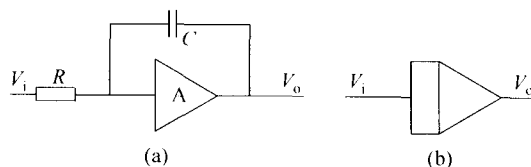


图3 积分器框图及符号图

可见输出电压 V_o 是输入电压 V_i 对时间的积分。积分的速度取决于时间常数 RC 。这就构成了积分器, 它是模拟计算机解常微分方程的最主要的基本运算部件。在图 2(b) 和图 3(b) 中, 还分别给出了加法器和积分器的符号图。标注在加法器输入端的数字表示比例系数 (R_f/R_i)。积分器也可以同时实现加法运算, 图 4 是具有加法功能的积分器的符号图。

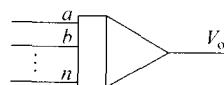


图4 具有加法功能的积分器符号图

模拟计算机的求解方法可以用一个例子来说明。例如对下面的微分方程求解:

$$\frac{d^2 x}{dt^2} + a \frac{dx}{dt} + bx = c$$

上式可以写成:

$$\frac{d^2 x}{dt^2} = -a \frac{dx}{dt} - bx + c$$

若不考虑初始条件, 上式可进一步写成:

$$\frac{dx}{dt} = \int_0^t \left(-a \frac{dx}{dt} - bx + c\right) dt$$

求解时, 通过排题板把反向器和积分器与上式对应地按图 5 所示连接起来, 就得到了 x , 即该微分方程的解。如果上述微分方程是对某一真实系统的描述, 则图 5 就是该系统的模拟。

利用特殊设计的函数发生器与运算放大器、比例加法器等相配合, 既能产生各种非线性函数, 也能实现如乘法、乘方、开方、除法等基本算术运算。例

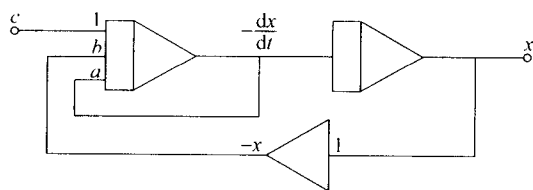


图5 一个真实系统的模拟

如,用两个平方函数发生器和3个加法器,可以构成一个乘法器,如图6(a)所示。图中FG代表函数发生器,其输出值为输入值的平方而符号相反(亦可相同)。作为输出的比例加法器,采用1:4(图中用0.25表示)的比例系数,所以,这种乘法器亦称四分之一平方乘法器,如图6(b)所示。于是最终输出为:

$$E_o = \frac{1}{4} [- (X + Y)^2 + (X - Y)^2] = -XY$$

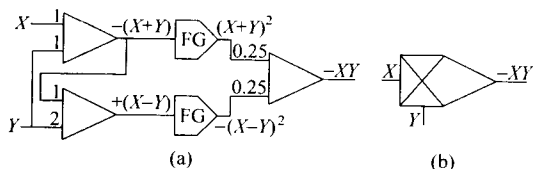


图6 四分之一平方乘法器框图及符号图

模拟计算机除运算部件外,还有接线排题板和相应的控制电路,用以连接各种运算部件并控制机器的启、停和其他操作。由于模拟计算机工作的连续性、并行性和实时性,而且操作简便,十分适用连续系统的实时仿真。其主要缺点是受元器件精度限制和运算放大器零点漂移的影响,整机的运算精度远低于数字计算机。可以把模拟计算机与数字计算机结合起来,组成混合计算机(参见混合计算机)。

参考文献

1. Korn G A, Korn T M. Electronic Analog Computers. New York: McGraw-Hill, 1956
2. Frederiksen T M. Intuitive Analog Electronics. New York: McGraw-Hill, 1989 (李人厚 童颖)

moshi

模式 (schema) 用数据定义语言定义的数据库结构。

一个数据库结构的定义包括要处理的实体类(同一类型实体的集合)及表征该实体类特征的各

属性,实体间的互相联系,以及数据库的完整性约束等。例如,职员、企业均是实体;职员与企业间有联系;进入企业的职员有年龄限制(约束)。关系数据库模式由一组关系模式组成。关系是一个由行和列构成的表,行代表一个实体而列代表属性。关系模式是关系的描述,指明表名、列定义和关系中数据应满足的条件(完整性约束)等。

美国国家标准局所属的 ANSI /X₃ /SPARC 的 DBMS(数据库管理系统)研究组曾于 1978 年提出了一般数据库系统的三级体系结构,十多年来一直为数据库界所沿用,该结构中把数据库分解为外模式、模式、内模式三级。外模式是给应用程序使用的部分,只反映整个数据库的一个局部;模式是对数据库整体结构的描述;内模式描述概念模式在物理存储器(磁盘、磁带等)上的实现。

参考文献

- Ullman J D. Principles of Database Systems (Second Edition). London: Pitman Publishing Ltd., 1987
(楼荣生 朱扬勇)

moshi fenleiqi

模式分类器 (pattern classifier) 对用以描述或表示事物和现象的数据进行分类的装置或过程。它是模式识别学科的重要组成部分。

智能行为的一个重要表现就是能够根据实际应用提出的要求,正确地把被识别对象划分到某一类别或某一范畴中去。模式分类正是在模式表示已经确定的情况下实现这种智能行为的机器方法。在更一般的意义下,模式分类可以看成是把大量的难以分析的数据(集)变换为较小的有实际应用意义的数据集的信息变换过程。自从 20 世纪 50 年代提出用统计决策方法实现模式分类以来,众多的学者对模式分类方法进行了理论和实际应用的研究,形成了统计的、句法的和人工神经网络的三个主要方向。其中基于统计决策理论和函数逼近理论的模式分类方法在理论上研究得最深入,应用面也最广泛(参见句法模式识别方法和人工神经网络在模式识别中的应用)。

为了使分类器具有良好的分类性能,首先要对分类器进行训练。最常见的一种训练方式是用一组特定的样本数据(观察)连同它所属的类别(称为训练样本集)作为输入,在一定的规则下,求出分类器的结构参数(训练阶段)。通过训练,分类器不断地改进自己的性能,从而具有对未知类别的模式进行

分类的能力(工作阶段)。这种训练过程又称作监督学习,或有教师学习。与此相对应的是只用一组类别未知的样本集合进行训练,在某些启发性知识指导下使分类器具有分类的功能。这种训练过程又称作非监督学习。它的主要方法是聚类分析。

用监督学习实现分类器设计可分为基于恢复概率密度函数的 Bayes 决策和直接求取分界面函数的两类方法。这两类方法都是用 n 维特征空间中的点(特征向量)表示属于各个类别的模式,并在一定意义上是在假设同一类别的模式在特征空间中集聚在一个或几个区域的条件下(紧致性假设)实现的。

Bayes 决策方法的基础是用同一类别的训练样本集恢复该类的概率密度函数 $p(x|\omega_i)$ 。其中 $\omega_i = 1, 2, \dots, C$, 表示 C 个类别, x 是表示模式的随机特征向量。如果待识别的模式特征向量为 X , 根据 Bayes 公式, 可求得 X 的后验概率

$$P(\omega_i | X) = \frac{p(X | \omega_i) P(\omega_i)}{\sum_{\omega_i=1}^C p(X | \omega_i) P(\omega_i)}$$

其中 $P(\omega_i)$ 是类别为 ω_i 的先验概率。Bayes 决策方法是比较各个类的 $P(\omega_i | X)$, 具有最大值的 ω_i 即为该模式所属的类别。理论上可以证明, Bayes 决策方法具有最小的误识率, 因此是一种最优分类器。如果考虑识别结果所造成的损失, 则可以设计具有最小损失的 Bayes 分类器。Bayes 决策方法需要预先确定各个类的先验概率并正确地恢复类条件下的概率密度函数。在概率密度函数的形式(例如高斯分布)已知的情况下, 可以用最大似然法或 Bayes 学习求出相应的分布参数, 在概率密度函数形式未知的情况下, 可以用 Parzen 窗技术求出概率密度函数(非参数法)。这种情况下, 往往需要大量的样本, 因此在很多实际问题中, 实现 Bayes 分类器有较大的困难。

直接求取分界面函数方法的基本思路是根据对问题性质的分析, 首先确定分界面的函数类(例如线性函数、二次多项式等等)然后在一定的准则下(例如训练集中样本被错分的数量最少), 求出其中的最优函数作为分界面, 从而把整个特征空间划分为若干区域, 落入到某个区域中的模式就分类为该区域所对应的类别。在最简单的两类分类情况下, 分界面函数可表示为 $f(x) = 0$, 若被识别模式的特征向量为 X , 则当 $f(X) > 0$ 时, X 属于第一类, 当 $f(X) < 0$ 时, X 属于第二类。函数类 $f(x)$ 的选择在很大程度上还取决于特征空间的维数和训练样本集

的样本数。当维数较高, 样本数又不多的情况下, 为了使分类器有较好的推广能力, 通常采用线性函数, 其相应的分界面则是超平面。

除上述两类方法外, 还可以用最近邻规则和位函数方法实现对模式的分类。

最近邻规则是对训练集中的样本进行筛选后, 直接存入计算机。计算被识别模式与这些样本的距离, 与被识别模式距离最小的样本所具有类别即为被识别模式的类别。理论上证明, 当训练集样本数接近无穷多时, 这种分类方法的分类性能接近最优分类器。已经发展了求取与被识别模式距离最小的样本的快速算法。

位函数方法是通过向分类器输入训练样本 X_k , $k = 1, 2, \dots, n$, 在函数 U 给定的条件下, 构造出两类的位函数, $U_1(x) = \sum_{X_k \in \omega_1} U(x, X_k)$, 和 $U_2(x) = \sum_{X_k \in \omega_2} U(x, X_k)$ 。取 $f(x) = U_1(x) - U_2(x)$ 作为分界面, 则当 $f(X) > 0$ 时, 把 X 分类为第一类, 当 $f(X) < 0$ 时把 X 分类为第二类。

参考文献

1. Sklansky J, Wassel G N 著. 模式分类器和可训练机器. 阎平凡等译. 北京: 科学出版社, 1987
2. 边肇祺等. 模式识别. 北京: 清华大学出版社, 1988 (边肇祺)

moshi shibie

模式识别 (pattern recognition) 对表征事物或现象的各种形式的(数值的, 文字的和逻辑关系的)信息进行处理和分析, 以便对事物或现象进行描述、辨认、分类和解释的过程。它是信息科学和人工智能的重要组成部分。

英文“pattern”源于法文“patron”, 本来是指可作为大家典范的理想的人, 或用以模仿复制的完美的样品。在模式识别学科中“模式”具有更广泛的含义。人们在观察事物或现象的时候, 常常要寻找它与其他事物或现象的相同或不同之处, 根据一定的目的把并不完全相同的事物或现象组成为一类。字符识别就是一个典型的例子。例如汉字“中”可以有各种写法, 但都属于同一类别。更为重要的是, 即使对于某个“中”的具体写法从未见过, 也能把它分到“中”这一类别。人们在路上行走的时候, 也总是不断地根据周围的景物, 判断他是否能达到目的地, 这实际上也是不断地在作“正确”与“不正确的

分类判断。人脑的这种思维能力就构成了“模式”的概念。在以上例子中,模式是和类别(集合)的概念分不开的,只要认识这个集合中的有限数量的事物或现象,就可以识别出属于这个集合的任意多的事物或现象。为了强调能从具体的事物或现象推断出总体,我们就把个别的事物或现象称作“模式”,而把总体称作类别或范畴。也有的学者认为应该把整个的类别称作模式,这样的模式是一种抽象化的概念,如“房屋”,“铁路”,“通俗音乐”等等都是模式,而把具体的对象如人民大会堂称作“房屋”这类模式中的一个样本。这种名词上的不同含义是容易从上下文中弄清楚的。

模式还可分成抽象的和具体的两种形式。前者如意识、思想、议论等,属于概念识别研究的范畴,是人工智能的另一研究分支。我们所指的模式识别主要是对语音波形、地震波、心电图、脑电图、图片、照片、文字、符号、三维物体和景物以及各种可以用物理的、化学的、生物的传感器对对象进行测量的具体模式进行分类和辨识。

模式识别研究主要集中在两方面,即研究生物体(包括人)是如何感知对象的,属于认知科学的范畴,以及在给定的任务下,如何用计算机实现模式识别的理论和方法。前者是生理学家、心理学家、生物学家和神经生理学家的研究内容,后者通过数学家、信息学专家和计算机科学工作者近几十年来的努力,已经取得了系统的研究成果。

早期的计算机模式识别研究着重在模型的建立上。20世纪50年代末,F. Rosenblatt提出了一种简化的模拟人脑进行识别的数学模型——感知机,初步实现了通过给定类别的各个样本对识别系统进行训练,使系统在学习完毕后具有对其他未知类别的模式进行正确分类的能力。60年代用统计决策理论求解模式识别问题得到了迅速的发展,70年代前后出版了一系列反映统计模式识别理论和方法的专著(参见**模式分类器**)。1962年,R. Narasimhan提出了一种基于基元关系的句法识别方法,傅京孙在这个领域进行了卓有成效的工作,形成了句法模式识别的系统理论(参见**句法模式识别方法**)。80年代,J. J. Hopfield深刻揭示出人工神经网络所具有的联想存储和计算能力,为模式识别技术提出了一种新的途径,短短几年在很多方面就取得了显著成果,从而形成了模式识别的人工神经网络方法的新的学科方向(参见**人工神经网络在模式识别中的应用**)。

一个计算机模式识别系统基本上是由三部分组

成的,即数据采集、数据处理和分类决策或模型匹配。任何一种模式识别方法都首先要通过各种传感器把被研究对象的各种物理变量转换为计算机可以接受的数值或符号(串)集合。习惯上,称这种数值或符号(串)所组成的空间为模式空间。为了从这些数值或符号(串)中抽取对识别有效的信息,必须对它进行处理,其中包括消除噪声,排除不相干的信号以及与对象的性质和采用的识别方法密切相关的特征的计算(如表征物体的形状、周长、面积等等)以及必要的变换(如为得到信号功率谱所进行的快速傅里叶变换)等。然后通过特征选择(参见**特征选择**)和提取或基元选择(参见**句法模式识别方法**)形成模式的特征空间。以后的模式分类或模型匹配就在特征空间的基础上进行。系统的输出或者是对象所属的类型或者是模型数据库中与对象最相似的模型编号。针对不同应用目的,这三部分的内容可以有很大的差别,特别是在数据处理和识别这两部分,为了提高识别结果的可靠性往往需要加入**知识库(规则)**以对可能产生的错误进行修正,或通过引入限制条件大大缩小待识别模式在模型库中的搜索空间,以减少匹配计算量。在某些具体应用中,如机器视觉,除了要给出被识别对象是什么物体外,还要求出该物体所处的位置和姿态以引导机器人的工作。

模式识别已经在天气预报、卫星航空图片解释、工业产品检测、字符识别、语音识别、指纹识别、医学图像分析等许多方面得到了成功的应用。所有这些应用都是和问题的性质密切不可分的,至今还没有发展成统一的、有效的可应用于所有的模式识别的理论。当前的一种普遍看法是不存在对所有模式识别问题都适用的单一模型和解决识别问题的单一技术,我们现在拥有的只是一个工具袋,我们所要做的是结合具体问题把统计的和句法(结构)的识别方法结合起来,把统计模式识别或句法模式识别与人工智能中的**启发式搜索**结合起来,把人工神经网络与各种已有技术以及人工智能中的**专家系统**,不确定推理方法结合起来,深入掌握各种工具的效能和应用的可能性,互相取长补短,开创模式识别应用的新局面。

参考文献

1. Watanabe S. Pattern Recognition; Human and Mechanical. New York; Wiley, 1985
2. 边肇祺等. 模式识别. 北京: 清华大学出版社, 1988

(边肇祺)

moshu zhuanhuanqi

模数转换器 (analog-to-digital converter, ADC)

将模拟信号转换成数字信号的设备。模数转换器是计算机用于工业控制时的重要输入设备。利用传感器将控制对象的温度、压力、流量、位移等物理量检测出来并转化成一定范围内连续变化的电流或电压模拟量,再通过模数转换器电路转换成二进制或十进制的数字量输入到计算机。模数转换器电路已有专用的芯片,采用的转换原理有逐次逼近式和双积分式等。逐次逼近式转换精度和速度都比较高,控制电路也不复杂,是 12 位以下模数转换器应用最广的一种。双积分式抗干扰能力强,转换精度高,但转换速度慢,因此多用于数字式测量仪表中。模数转换器实际上是由采样保持器、多路开关和模数转换器电路组成的一种计算机输入通道。它有下列几种结构形式:

(1) 不带采样保持器的单通道(图 1) 这种结构常用于低频或直流信号。

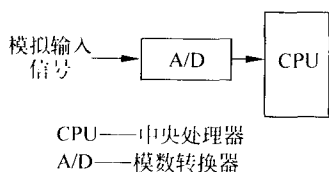


图 1 单通道

(2) 带采样保持器的单通道(图 2) 适用于模拟信号时间变化率较大的情况。与上一种方式不同的是增加了采样保持器 S/H。

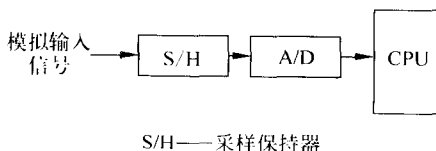


图 2 单通道

(3) 每路有单独采样保持器的多通道(图 3) 这种结构用于高速系统,各通道独立进行转换,互不干扰,缺点是所需硬件多。除了采样保持器 S/H 及模数转换器 A/D 外,每个通道要增加输入输出接口电路 I/O。

(4) 多路共享 ADC 的通道(图 4) 这种结构的比上一种慢,因为信号要经过多路开关进行转换,串行进入 A/D 转换器,但各路采样保持器是

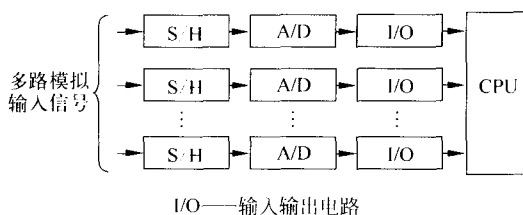


图 3 多通道 A/D 转换

独立的。

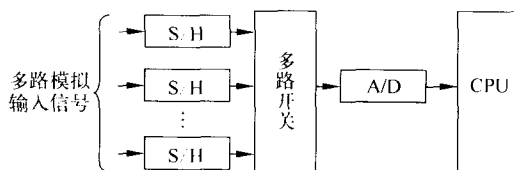


图 4 多通道共享 A/D 转换器

(5) 多路共享采样保持器的通道(如图 5) 这种结构硬件最省。

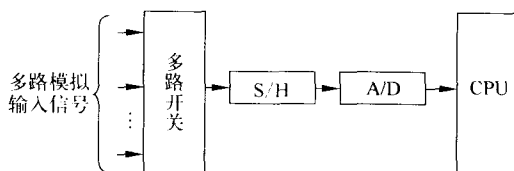


图 5 多通道共享采样保持器与 A/D 转换器

用于微型计算机的 A/D 通道插件已有商品销售。一个插件上有 4 路、8 路等。A/D 通道特性用每秒采样数(采样频率)和输出精度(数字化量的位数)来表示。例如:CD-ROM 机的 ADC 通道,采样频率为 48 kHz,数字化位数为 16 位。A/D 通道需要有软件支持,在硬件设计的同时要考虑必要的驱动程序。

参考文献

1. 王秀玲等. 微型计算机 A/D, D/A 转换接口技术及数据采集系统设计. 北京: 清华大学出版社, 1984
2. 刘乐善等. 微型计算机接口技术原理及应用. 武汉: 华中理工大学出版社, 1996 (林兼)

motai luoji

模态逻辑 (modal logic) 在非经典逻辑中, 考察“必然”、“可能”、“偶然”等模态概念的逻辑性

质,研究模态命题之间、模态命题与非模态命题之间形式推理的一个分支学科。模态逻辑的内容包括模态逻辑系统(不同的系统体现了对“必然性”概念的不同理解),语义理论,模态逻辑系统的元逻辑特性(例如:一致性、可靠性、完全性和可判定性),以及模态逻辑与现代逻辑其他分支学科的相互关系等。

模态命题逻辑系统 最基本的模态命题逻辑系统是 T, S_4, B 和 S_5 。其中系统 T 可以用如下的方式表述:

(1) 初始符号 ① p_1, p_2, \dots ; 它们可解释为可数个命题变元。② $\neg, \rightarrow, L, (,)$; 它们可分别解释为否定词、实质蕴涵、必然算子、左括号和右括号。

(2) 形成规则 ① 每一个命题变元 $p_i (i = 1, 2, \dots)$ 是公式; ② 若 A, B 是公式, 则 $\neg A, (A \rightarrow B)$ 和 LA 是公式。

(3) 公理 (A, B, C 是任意的公式) ① $A \rightarrow (B \rightarrow A)$; ② $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$; ③ $(\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A)$; ④ $LA \rightarrow A$; ⑤ $L(A \rightarrow B) \rightarrow (LA \rightarrow LB)$ 。

(4) 推理规则 (A, B 是任意的公式) ① 分离规则: 由 A 和 $A \rightarrow B$ 可推出 B 。② 必然规则: 由 A 可推出 LA 。

在系统 T 的基础上, 再添加公理 $LA \rightarrow LLA$ 即构成系统 S_4 。对系统 T 分别添加公理 $M LA \rightarrow A$ 和 $MA \rightarrow L MA$ (M 为“可能算子”, 其定义为 $\neg L \neg$) 则分别构成系统 B 和系统 S_5 。这 4 个系统都是正规系统。系统 T 中没有任何归约律, S_4 中有两种归约律 $LA \leftrightarrow L LA$ 和 $MA \leftrightarrow M MA$, S_5 中则有全部 4 种归约律成立(另两种是 $MA \leftrightarrow L MA$ 和 $LA \leftrightarrow M LA$)。因而在 S_5 中任何高于 1 级的公式都可以归约为 1 级公式。对系统 B 可以作某种直觉主义的解释, 即把 $\neg M$ 解释为直觉主义否定。

在非正规系统中, 受到逻辑学家关注的有 $S_{0.5}, S_2, S_3, S_{3.5}$ 等。

模态谓词逻辑系统 最基本的一阶模态谓词逻辑系统是 $T + BF$ 和 $LPC + T$, 它们都是在模态命题逻辑系统 T 和经典的一阶谓词逻辑系统 LPC 的基础上建立起来的。其中系统 $T + BF$ 可以用如下的方式表述:

(1) 初始符号 ① x_1, x_2, \dots ; 它们可解释为可数个个体变元。② $A_1^1, A_1^2, \dots, A_1^i, A_2^1, A_2^2, \dots, A_2^i, \dots$; 它们可解释为谓词字母, A_i^j 是 i 元谓词。③ $\neg, \rightarrow, L, \forall, \dots, (,)$; 其中 \forall 是全称量词。

(2) 形成规则 ① $A_i^j(t_1, t_2, \dots, t_i) (i = 1, 2, \dots; j = 1, 2, \dots)$ 是公式, 其中 t_1, t_2, \dots, t_i 是任意的个体变元。② 若 A, B 是公式, 则 $\neg A, (A \rightarrow B), LA$ 和 $(\forall x_i) A (i = 1, 2, \dots)$ 是公式。

(3) 公理 (A, B, C 是任意的公式) ① ~ ⑤ 形式同系统 T 的公理 ① ~ ⑤。⑥ $(\forall x_i) A \rightarrow A(x_j)$, 其中 x_j 对于 A 中的 x_i 是自由的, 用 x_j 取代 A 中 x_i 的每一次自由出现所得的公式是 $A(x_j)$ 。⑦ $(\forall x_i) (A \rightarrow B) \rightarrow (A \rightarrow (\forall x_i) B)$, 其中 x_i 在 A 中没有自由的出现。⑧ $(\forall x_i) LA \rightarrow L(\forall x_i) A$ 。

(4) 推理规则 (A, B 是任意的公式) ① 和 ② 同系统 T 的分离规则和必然规则。③ 概括规则: 由 A 可推出 $(\forall x_i) A (i = 1, 2, \dots)$ 。

系统 $T + BF$ 中的公理 ⑧ 称为巴坎公式。若在系统 $T + BF$ 的公理中去除巴坎公式, 即构成系统 $LPC + T$ 。巴坎公式亦可表述为 $(\forall x_i) LA \leftrightarrow L(\forall x_i) A$ 或 $(\exists x_i) MA \leftrightarrow M(\exists x_i) A$, 它体现了全称量词和必然算子, 存在量词和可能算子的可交换性。包含巴坎公式的系统与不包含巴坎公式的系统有完全不同的模型特征。受到关注的其他一阶模态逻辑系统有 $S_4 + BF, LPC + S_4, LPC + B, LPC + S_5$ 以及某些比 $LPC + T$ 更弱的模态谓词逻辑系统。

语义理论 主要有克里普克语义和代数语义, 前者更为重要。模态命题逻辑系统的克里普克语义模型的一般形式为有序三元组, 其中系统 T 的模型可以用如下方式表述:

一个 T 模型是一个有序三元组 $\langle W, R, V \rangle$, W 是非空集, 称为可能世界集; R 是 W 上的一个二元自反关系, 即 $R \subseteq W \times W$, 并且对每一个 $w_i \in W$, 都有 $w_i R w_i$; V 是赋值函数, 其定义域是 $F \times W$, 其中 F 为系统 T 中全体公式组成的集合, 值域是 $\{1, 0\}$, 1 表示“真”, 0 表示“假”, 并且 V 满足以下 4 个条件:

(1) 对每一个命题变元 p_i 和每一个可能世界 $w_i \in W$, 有 $V(p_i, w_i) = 1$ 或 $V(p_i, w_i) = 0$, 但不同时成立。

(2) 对每一个公式 A 和每一个 $w_i \in W$, 有 $V(\neg A, w_i) = 1$, 当且仅当 $V(A, w_i) = 0$ 。

(3) 对任意的公式 A, B 和每一个 $w_i \in W$, 有 $V(A \rightarrow B, w_i) = 1$, 当且仅当 $V(A, w_i) = 0$ 或 $V(B, w_i) = 1$ 。

(4) 对任意的公式 A 和每一个 $w_i \in W$, 有 $V(LA, w_i) = 1$, 当且仅当对每一个满足条件 $w_i R w_j$ 的 $w_j \in W$, 都有 $V(A, w_j) = 1$ 。

当一个公式 A 在每一个 T 模型的每一个可能世

界中都为真时,称公式 A 是 T 有效的。可以证明 A 为 T 有效的充分必要条件是 A 为 T 中的定理,即系统 T 具有可靠性和完全性。对 T 模型的二元关系 R 分别添加传递性(即若 $w_i R w_j$ 且 $w_j R w_k$,必有 $w_i R w_k$)和对称性(即若 $w_i R w_j$,则有 $w_j R w_i$),则分别构成 S_4 模型和 B 模型。当 R 同时具有自反性、传递性和对称性时,为 S_5 模型。系统 S_4, B, S_5 都具有可靠性和完全性。借助于克里普克模型,还可以证明系统 T, S_4, B, S_5 都是能行可判定的。

模态谓词逻辑系统的克里普克模型要涉及到个体域 D 。包含巴坎公式的系统 $T + BF$ 和 $S_4 + BF$ 的模型结构为有序四元组 $\langle W, R, D, V \rangle$,它的特点是:每个可能世界有着相同的对象域 D ;然而,不同的可能世界中,对象的性质或对象间的相互关系则可以不同。而不包含巴坎公式的系统 $LPC + T$ 和 $LPC + S_4$ 的模型结构为有序五元组 $\langle W, R, D, Q, V \rangle$,其中 Q 是由 W 到 D 的幂集的一个函数,它为每一个可能世界 $w_i (\in W)$ 指定了一个对象域 $D_i (\subseteq D)$,即允许不同的可能世界有不同的对象域。

模态逻辑的研究成果促进了广义模态逻辑的时态逻辑、道义逻辑等分支学科的发展,克里普克的可能世界语义理论已被广泛应用于众多的现代逻辑分支学科以及自然语言逻辑的研究之中,模态逻辑在人工智能研究中的重要性也日益显现出来。

参考文献

1. Hughes G E and Cresswell M J. An Introduction to Modal Logic, Second edition. 1972
2. Gabbay D and Guenther F. Handbook of Philosophical Logic, Vol II. D. Reidel Publishing Company, 1984 (冯棉)

moxing jianyan

模型检验(model checking) 通过搜索待验证(软件或硬件)系统模型的有穷状态空间来检验系统的行为是否具备预期性质的一种自动验证技术。

模型检验最初是由 EM Clarke 和 EA Emerson 在 1981 年提出的,他们设计了检验有穷状态迁移系统是否满足给定 CTL 公式的算法。JP Quille 和 J Sifakis 在同年也提出了类似的思想。在模型检验中,系统用有穷状态模型建模;其性质规约通常是时序逻辑或模态逻辑公式,也可以用自动机语言描述;通过有效的搜索来检验有穷状态模型是否满足规约,如果不满足,它还能给出使性质公式为假的系统

行为轨迹。1993 年 KL McMillan 基于 RE Bryant 的有序二叉判定图(OBDD)来有效地表示状态迁移系统,提出了符号化模型检验,大大提高了可有效应用模型检验技术的系统规模,使得模型检验在工业界逐步得到应用。模型检验已经成为形式化方法中系统验证的重要途径。

模型检验应用面临的主要问题是状态爆炸问题。由于系统的有穷状态模型的状态数量往往随其模型的并发分量的增加呈指数增长,因此,复杂系统建模时其可达的状态空间常常难于在计算机存储器中全部构建,也就无法进行模型检验了。一种途径是通过发现模型的状态空间的结构特点来缓解状态空间爆炸问题,主要的方法包括符号化模型检验、对称模型检验、偏序模型检验、On-the-fly 模型检验等。在符号化模型检验时,用布尔公式刻画状态集合和状态对集合,用 OBDD 来表示这些布尔公式,使用 OBDD 上的布尔操作来计算谓词转换子(其不动点刻画了 CTL 模式子),从而使模型检验在压缩了的符号化状态空间上来验证 CTL 性质。对称模型检验针对由多个完全类似的进程组成的系统,利用其模型的状态空间的对称性来生成压缩的且对模型检验等价的模型。偏序模型检验通过发掘系统中并发执行的迁移的交换性,减少本质上相同的状态,从而仅生成足以检验预期性质的小部分状态空间。On-the-fly 模型检验把状态空间生成和检验它是否满足性质合在一起进行,而不去预先生成整个状态空间,从而尽可能避免状态爆炸。另一种途径是通过抽象和分解把复杂系统的验证转化成模型检验可以处理的问题,主要的方法包括抽象方法、组合方法等。抽象方法通过去掉原来模型中与待验证性质无关的信息而获得简化模型的方式来减小模型检验时问题的规模。抽象必须满足:若简化的模型具备某性质,则原来的模型亦具备该性质。组合方法基于分而治之的思路来缩减模型检验时问题的规模,先验证系统构件的局部性质,然后把这些性质组合起来获得系统的全局性质。

模型检验在硬件设计和通信协议的形式验证上获得了巨大的成功。例如,利用模型检验技术,有效地发现了多处理器高速缓冲存储器一致性协议中的一些错误,而这些错误在传统的模拟中往往未能发现。著名的基于模型检验的并发系统自动验证工具有 SMV、SPIN 等。2001 年 G. Holzmann 研发的模型检验系统 SPIN 获 ACM 软件系统奖。

模型检验技术仍在发展,主要方向是从系统的

建模模型和性质规约语言的角度扩展模型检验,例如,模型检验解决实时和混合系统、软件系统的验证问题、一阶模态逻辑模型检验;与其他软件技术的结合,如模型检验与软件测试,与基于定理证明的形式验证技术的结合。

参考文献

1. McMillan K L, Model Checking. Encyclopedia of Computer Science. 4th edition, Macmillan Reference Ltd, 2000
2. Clarke Edmund and Holger Schlingloff. Model Checking. In: Handbook of Automated Reasoning, Edited by Alan Robinson and Andrei Voronkov, Elsevier Science Publishers, 2001
3. 林惠民, 张文辉. 模型检测: 理论、方法与应用. 电子学报, 2002, 30(12A) (王戟)

moxinglun

模型论 (model theory) 研究形式系统及其解释之间关系的理论。在 20 世纪 20 年代, T. Skolem 等人在数理逻辑研究中就得到模型论性质的重要结果。但模型论形成系统的理论, 大致在 20 世纪 50 年代, 其奠基人应推 A. Tarski, A. Robinson 也作出了很大贡献。

一个形式语言的解释称为此语言的一个结构(模型)。一阶语言的结构是一个具有若干函数、关系和特指元素的集合(参见一阶逻辑), 也称为泛代数。所以, 模型论被形容为“泛代数+逻辑”。由于所涉及的逻辑系统不同, 模型论又可分为: 一阶模型论、高阶模型论、无穷长语言模型论、模态模型论、多值模型论等。由于在数理逻辑中以一阶逻辑发展最成熟, 所以, 模型论也以一阶模型论内容最丰富, 应用最广泛。

如果一阶理论 T_1 的非逻辑公理集是一阶理论 T_2 的非逻辑公理集的有穷子集, 则称 T_1 是 T_2 的有穷公理化部分。紧致性定理指出: 如果一阶理论 T 的每个有穷公理化部分有模型, 则 T 有模型。它的另一等价形式是: 如果公式 A 在一阶理论 T 中有效, 则 A 在 T 的某个有穷公理化部分中有效。紧致性定理的应用很广, 可以用它讨论一阶逻辑的表达能力, 证明某些概念不能用一阶逻辑表达。非标准分析的基础是实数系的非标准模型, 其存在性的证明就使用了紧致性定理。

通常把论域的基数称为结构的基数, 把一阶理论的公式集的基数称为该理论的基数。勒文海姆-

斯拜伦定理指出, 有模型的可数理论必有可数模型。后来, Tarski 将其推广为基数定理。设 λ 是无穷基数, 如果基数为 λ 的理论有无穷模型, 则它有基数为任何 $\alpha \geq \lambda$ 的模型。这个定理在模型论及公理集合论中常被使用。

不出现量词的公式称为开公式。设 U 和 B 是一阶语言 L 的结构。如果 U 的论域是 B 的论域的子集, 并且对于每个开公式 A , 只要使 A 中的自由变元指定 U 中个体为值, A 在 U 和 B 中的意义总是相同, 就称 U 是 B 的子结构。如果将上述定义中的“每个开公式”改为“每个公式”, 则成为初等子结构的定义。例如, $\langle \mathbf{Z}; < \rangle$ 是 $\langle \mathbf{Q}; < \rangle$ 的子结构, $\langle \mathbf{Q}; < \rangle$ 是 $\langle \mathbf{R}; < \rangle$ 的初等子结构, 其中 $\mathbf{Z}, \mathbf{Q}, \mathbf{R}$ 分别表示整数集, 有理数集, 实数集, “ $<$ ”是小于关系。

设 U 和 B 是一阶语言 L 的结构, ψ 是 U 的论域到 B 的论域的双射, 如果 ψ 将 U 中每个个体映射到 B 中有同样性质的个体, 则称 ψ 是 U 到 B 的同构。如果存在 U 到 B 的同构, 则称 U 和 B 同构。同构的结构其逻辑性质完全相同, 可将它们看做同一个结构。

称定理相同的理论为等价的理论, 如果对于每个闭公式 A , A 和 $\neg A$ 之中恰有一个是理论 T 的定理, 则称 T 是完全的理论。如果结构 U 和 B 有完全相同的一阶性质, 则对于每个闭公式 A , $U \models A$ 当且仅当 $B \models A$, 就称 U 与 B 初等等价。以在结构 U 中有效的闭公式为非逻辑公理的理论称为 U 的理论, 记为 $Th(U)$ 。对于一个有模型的理论 T 来说, 以下 3 个条件是等价的: ① T 是完全的理论; ② T 的任何两个模型初等等价; ③ T 等价于 $Th(U)$, 其中 U 是 T 的一个模型。

如果理论 T 的每个模型 U 的子模型都是 U 的初等子结构, 则称 T 为模型完全的理论。如果理论 T 的模型 U 同构于 T 的每个模型的一个子结构, 则称 U 为 T 的素模型。一个有素模型的模型完全的理论必是完全的理论。例如, 特征为 0 的代数闭域理论是模型完全的理论, 并且代数数域是它的一个素模型, 因此该理论是完全的理论。

只有一个模型的理论称为范畴的理论。 T 是范畴的理论当且仅当 T 是仅有有穷模型的完全理论。范畴性的概念对理论的限制太严了, 排除了一切具有无穷模型的理论, 而常用的数学理论许多都是有无穷模型的。因此, 在模型论中又引进了 α 范畴的概念。设 α 是一个基数, 如果理论 T 只有一个基数为 α 的模型, 则称 T 为 α 范畴的理论。关于范畴性

的一个著名定理是:如果可数理论 T 对于一个不可数基数 α 是 α 范畴的,则 T 对于任何不可数基数 β 也是 β 范畴的。

由理论的 α 范畴性可以得出它的完全性。设 T 是基数为 α 的没有有穷模型的理论,基数 $\beta \geq \alpha$ 。如果 T 是 β 范畴的理论,则 T 是完全的理论。例如,无最大元和最小元的稠密全序理论是 \aleph_0 范畴的可数理论。它只有无穷模型。因此,它是完全的理论。

模型论与数理逻辑的其他分支有着密切的联系。首先,各种逻辑演算是模型论的基础。此外,在证明论中,有关判定问题的研究广泛使用着模型论方法。在公理集合论中,有关大基数的研究与模型论有密切的联系。另外,布尔值模型被应用于各种独立性问题的研究。公理集合论中的力迫法被移植于模型论中。

模型论与抽象代数、数学分析、数论、拓扑学等数学学科也有联系。模型论中的概念和方法有不少

来源于泛代数。模型论的方法被用于解决抽象代数中的某些问题。由鲁滨逊创立的非标准分析则是模型论与数学分析相结合的产物。

模型论的概念和方法也应用于计算机科学和人工智能。实际上,程序设计语言就是一种形式语言,在程序设计语言的语义研究中涉及到许多模型的概念。抽象数据类型的代数语义就是等式逻辑的模型。在非单调逻辑的形式系统及其语义研究中,大量地使用了模型论的概念和方法。

参考文献

1. Chang C C, Keisler H J. Model Theory. Third edition. Amsterdam; North - Holland, 1990
2. Bridge J. Beginning Model Theory. Oxford: Clarendon Press, 1977
3. Enderton H B. A Mathematical Introduction to Logic. New York: Academic Perss, 1972 (何自强)

N

neibu luyou xieyi

内部路由协议 (interior routing protocol)

在一个自治系统 (AS) 内部使用的路由选择协议。大型的复杂计算机网络由许多自治系统组成。每个自治系统内部有自己公共的管理部门和路由策略。在一个自治系统内部使用的路由协议就是内部路由协议。它是相对于自治系统之间采用的外部路由协议而言的。

早期 ARPA 网中采用的 **RIP 协议**、国际标准化组织 ISO 采纳的 IS-IS 协议以及 Internet 中采用的开放最短路径优先 **OSPF 协议** 都属于内部路由协议。直到 1979 年, ARPA 网中使用的都是 **RIP 路由协议**, 这是一种距离向量协议。后来虽然有了新的链路状态协议, 但 **RIP** 仍在使用。IS-IS 和 **OSPF** 都是链路状态协议。此外, 还有专用网络间接口 (PNNI) 和 NetWare 链路服务协议 (NLSP) 也是采用链路状态算法的内部路由协议。PNNI 专用于异步传送模式 (ATM) 网络中, 而 NLSP 则专用于 NetWare 中。**RIP** 和 **OSPF** 由于得到 Internet 的 IETF 支持而使用最为广泛。目前大多数路由器产品都支持这两个内部路由协议。

参考文献

1. Stallings W. Data and Computer communications. 5th Edition. New Jersey: Prentice Hall, 1997
2. Shay W A. 数据通信与网络教程. 高传善等译. 北京: 机械工业出版社, 2000 (高传善)

neilianwang

内联网 (Intranet) 基于 TCP/IP 协议, 使用万维网 (WWW) 工具, 采用防止外部侵入的安全措施与外界连接, 为企业或单位内部服务的企业内部网络。

从这个定义出发, 可概括出内联网的如下 5 个要点:

(1) 内联网是根据企业内部的需求而设置的, 它的规模和功能是根据企业经营和发展的需求确定的;

(2) 内联网不是一个孤岛, 它能方便地和外界连接;

(3) 内联网采用 TCP/IP 协议 (参见 **TCP/IP 协议集**) 及相应的技术和工具, 是一个开放的系统;

(4) 内联网根据企业的安全要求, 设置相应的防火墙、安全代理等, 以保护企业内部的信息, 防止外界侵入;

(5) 内联网广泛使用万维网 (WWW) 的工具, 使企业员工和用户能方便地浏览和采掘企业内部的信息以及 Internet 的丰富的信息资源。这些工具包括超文本置标语言 (HTML)、公共网关接口 (CGI) 以及新的编程语言 **Java 语言** 等。

组建一个内联网的主要任务是:

- (1) 网络基础设施的需求分析和开发;
- (2) 安全的需求分析和实施;
- (3) 对 **Internet 服务提供者** 的评估和选择;
- (4) 软、硬件的选择和安装, 包括服务器、浏览器、搜索工具、文本写作工具、文本转换工具、文本数据库以及数据库查询工具等;
- (5) 内联网的维护。

内联网是企业适应网络化经济发展需求的重要信息基础设施。新的联网的组织结构不只是简单的面向处理的组织, 也不只是简单的基于工作组的结构, 而是从根本上重新构思组织的功能和性质以及组织之间的关系。这种新的组织被称为 **互联网络的企业**, 这是包括各个层次和经营功能的巨大的关系网。它使企业能有效地获得资源, 使企业变成由更小的分子族组成, 且能很方便、很灵活地协同工作。互联网络的企业将扩展成虚拟的企业, 可经常重构经营的关系, 如同 Internet 一样, 企业的每个成员可方便地介入, 且为企业做更多的贡献。

参考文献

1. 胡道元. INTRANET 网络技术及应用. 北京: 清华大学出版社, 1998
2. Douglas E Comer. Internetworking with TCP/IP, Vol. I, 3rd ed. Prentice Hall Inc., 1995

(胡道元)

nengli chengshudu moxing

能力成熟度模型 (capability maturity model, CMM) 描述和分析软件机构的软件过程能力和成熟程度以建立其分级标准和框架的模型。软件过程能力是指描述遵循软件过程而得到所期望结果的程度;软件过程成熟度是指软件机构中软件过程被明确定义、管理、控制及其实效的程度。利用 CMM 模型,软件机构可以通过评估自己当前的过程成熟程度,来选择相应的改进策略,以达到更高一级的成熟程度;采购机构可以确定与软件机构签订合同的有关风险,并帮助管理正在进行中的合同。

在 20 世纪 30 年代,贝尔实验室物理专家 Walter Shewart 提出了统计质量控制原理。W. Edwards Deming 和 Joseph Juran 在 40—50 年代发展了这些原理,并在实践中得到证明。20 世纪 70 年代末期,ITT 的 Philip Crosby 把这些原理用于构造成熟度框架,并首次提出了质量实践的五个进化阶段。

20 世纪 80 年代中期,IBM 在 Watts S. Humphrey 的指导下,Ron Radice 等人将此成熟度框架首次用于软件过程,并由 Humphrey 于 1986 年将此成熟度框架带到卡内基·梅隆大学的软件工程研究所 (CMU/SEI)。1987 年 9 月,CMU/SEI 发表了一个简短的软件过程成熟度框架。其后在 Humphrey 的

“管理软件过程”一书中进行扩充。在此基础上,1990 年 Jim Withey, Mark Paulk 以及 Cynthia Wise 提出了最早的草案。1991 年 8 月 Mary Beth Chrissis 和 Bill Curtis 帮助 Mark Paulk 校订,提出了 CMM1.1 版。

由于美国联邦政府的大力推行,能力成熟度模型 (CMM) 得到了广泛应用。据 CMU/SEI 于 2001 年公布的统计,通过 CMM 评估的软件机构共有 964 家。目前 CMM 本身正在向纵深发展,能力成熟度模型族包括:

- 软件能力成熟度模型 (SW-CMM)
- 软件获取能力成熟度模型 (SA-CMM)
- 人员能力成熟度模型 (People-CMM)
- 系统工程能力成熟度模型 (SE-CMM)
- 集成产品开发能力成熟度模型 (IPD-CMM)
- 个体软件过程 (PSP)
- 群组软件过程 (TSP)

另外,国际标准化组织为制订软件过程评价标准,成立了“软件过程改进和能力确定 (SPICE)”项目,ISO/IEC JTC1 的 SC 7 分技术委员会于 1996 年 9 月正式公布了工作草案,代号为 15504。

能力成熟度模型的组成主要包括过程能力、成熟度级别、关键过程区域、目标、关键实践、关键指示因子和提问单等。CMM 的结构如图 1 所示。

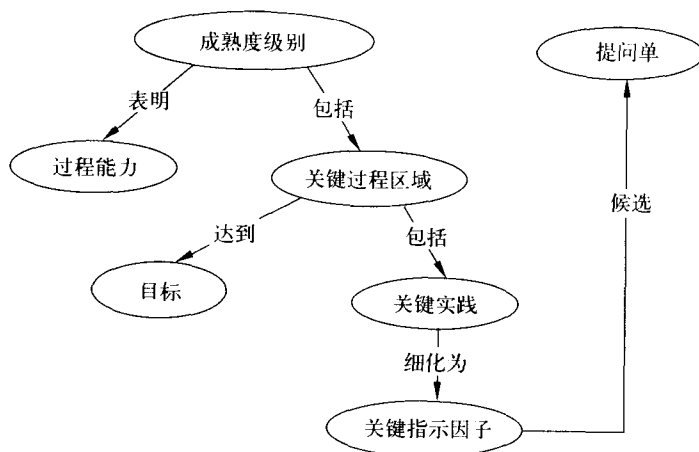


图 1 能力成熟度模型的结构

成熟度级别 是软件机构向成熟方向迈进的台阶,CMM 有五个成熟度级别。

关键过程区域 是一组信息。它指明了一个软件机构为改进其软件过程所应集中关注的区域。实

践证明它对改进过程能力最为有效,同时也指明达到一个成熟度级别所必须着手解决的问题和必须满足的要求。CMM 共有 18 个关键过程区域,它分别划归于 1 级除外的其他四个成熟度级别中。

目标 概括和表明了关键过程区域的范围、界限、目的。当该关键过程区域的目标已达到,可以说,以该关键过程区域为特征的过程能力已被规范化了。

关键实践 是对一个关键过程区域的具体化和细节化的描述,为软件机构的过程改进工作提供了具体的指导。当完成关键实践时,将能达到那个关键过程区域的目标。CMM 模型包含多达 316 个关键实践,其描述形式具有相同的特征,即以目标、承

诺、能力、活动、监控和验证为标志,分类进行具体、细化的描述。

关键指示因子 是过程实践的组成部分,其主要作用是帮助鉴别一个关键过程区域目标是否已经达到。

提问单 列出了有关软件过程的一组“是-否”的提问,它是对每一个关键过程区域的实践的采样,适用于对软件过程能力进行调查研究。

CMM 的能力成熟度级别如图 2 所示。

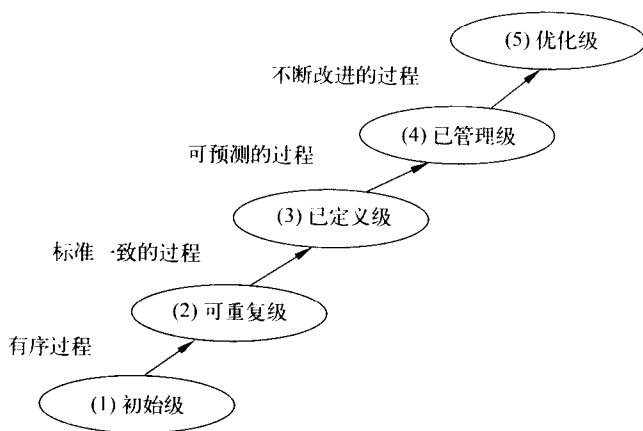


图2 能力成熟度模型的能力成熟度级别

1级——初始级 处于1级的软件机构,其过程能力是不可预测的,软件项目所涉及的进度、预算、产品功能和产品质量等一般是不可预测的,产品性能只能根据相关人员的个人能力而不是机构的过程能力来预测。

2级——可重复级 处于2级的软件机构的过程能力可以概括为软件项目的策划和跟踪是稳定的。一个有序的管理过程提供了可重复以前成功实践的项目环境。

3级——已定义级 处于3级的软件机构的过程能力可以概括为无论是管理活动还是工程活动都是稳定的。在已建立的产品生产线上,成本、进度、功能和质量都受到控制,而且软件产品的质量具有可追溯性。

4级——已管理级 处于4级的软件机构的过程能力可以概括为软件过程是可度量的。这一级过程能力表明软件机构在定量的范围内预测过程和产品质量趋势,当发生偏离时,能及时采取措施予以纠正,并可预测软件产品是高质量的。

5级——优化级 处于5级的软件机构的过程能力可以概括为软件过程是可优化的,能够持续不断地改进其过程能力,也能借助新技术和新方法来实

现未来的过程改进。

能力成熟度模型的广泛采纳和成功推广也推动了软件以外其他相关学科类似模型的开发。各种模型的推行,出现了许多问题,如过程改善目标和技术冲突,对培训工作的需求有较大的增长,部分实践人员在应用各种不同的模型来实现特定的需要时产生了混淆等。针对这种情况,美国联邦政府、产业界和CMU/SEI于1998年启动了“能力成熟度模型集成(CMMI)”项目,于2000年第四季度发布了第一个正式的CMMI,最近的修改稿是2002年6月10日发布的。CMMI包括了大量的工程改善和过程改善的信息,提供了单一的集成化框架来改善跨学科的机构的工程过程,从而提高机构过程改善的质量和有效性。相信CMMI将会逐步替代CMM。

参考文献

1. Paulk M C, Curtis B, Chrissis M B and Weber C. V. Capability Maturity Model for Software. CMU/SEI-93-TR-24, 1992
2. ISO/IEC TR 15504. Information technology-software process assessment. 1996
3. <http://www.sei.cmu.edu/cmmi> (朱三元)

P

paiban ruanjian

排版软件 (page layout software) 将文字、图形和图像等合理安排在页面内的软件。排版软件在安排文字时,必须处理文字排版的各种要求,包括字体、字号的变化,英文换行时的拆音节处理,各种禁排的处理(如标点符号不能排在行首)等。排版的目的是使排出来的版面既要美观,又要符合国家标准。排版软件不仅要在版面中安排文字,还要安排图像及画一些图形,优秀的排版软件还能对文字、图像进行旋转、倾斜等操作。

排版软件是电子印前处理系统的重要组成部分。它与图像处理和图形软件结合使用,可以设计各种各样复杂的版面,并可直接打印输出或者生成页面描述语言描述的版面。

排版软件按操作处理的方式可以分为两类。

批处理方式排版软件 它把描述页面的命令和数据录入到文件里,再通过排版软件解释这些命令生成版面。在生成版面的过程中,用户与计算机软件没有交互,而是由计算机一次完成。由于计算机软件 and 硬件环境的限制,早期的排版软件都是批处理方式。国外的西文批处理排版软件有 TEX 等,国内的中文批处理排版软件有北大方正书版软件等。TEX 是优秀的排版软件,特别对数学公式的排版质量较好。北大方正书版软件采用方正书刊排版语言,它除了能排汉字外,还能排数学公式、表格等,同时具有自动生成页号、书眉等功能。批处理方式适合于比较简单、变化不多的版面排版,如小说、科技论文和词典等,对于报纸、广告等复杂版面的排版不很适用。

交互式排版软件 随着个人计算机的日益普及,出现了很多交互式排版软件,这些软件具有所见即所得的功能。在制作版面时,显示器显示的结果与实际打印输出的结果是一致的,并且用户在制作版面的同时就能看到显示的结果。一个所见即所得的排版软件,它能以各种字体和字号显示文字,并能以各种色彩显示图像,但由于显示器的分辨率及色彩与实际输出设备不可能完全一致,因此屏幕显示结果与输出的结果不可能精确吻合。另外,交互式系

统要求各种操作的响应时间短,所以显示时可能要降低一些精度。实际上,交互式排版软件系统要求显示的结果尽量与输出的结果一致。

排版语言是一种专门用来排版的计算机语言,它由一系列特定的、描述版式的命令构成。主要用于批处理排版方式。操作员根据版式要求,选择相关的命令,将其插入到需要排版的文字中,由排版程序转换生成排版结果(即版面信息)。

排版语言一般具有如下特点:

(1) 既提供标题、表格、分栏、对照等高层命令,表达结构性信息,又提供如字体、字号、行距、空格、基线等低层命令,指定具体的排版要求;

(2) 命令中有一个或多个参数可供选择,并提供默认值;

(3) 能够处理多种复杂版式,如版面划分、放置图片、数学式、化学式以及表格等;

(4) 提供自动生成书眉、页码、脚注、词条和目录等功能;

(5) 用户几乎可以在任意位置排任意复杂的版式,有很大的灵活性;

(6) 提供宏定义功能,从而简化了排版中固定格式的使用;

(7) 命令名一般是由命令标识符加上其他字符组成,容易记忆,使用和处理简便。

目前的排版语言在描述版面方面的功能已相当完善,但还应加强对文献结构信息(即构成一本书或一篇文章的各种类型的元素,如章节、段、内容提要、注释、插图、有序列表等)的描述。这样的描述有利于多个系统之间的数据交换与共享,能提高出版物的规范化程度,便于对已排内容的修改和再版,还可用于排版系统与其他信息处理系统的连接等。

参考文献

1. S D Peter. An Introduction to Text Processing. MIT Press, 1990
2. Computer Dictionary. Microsoft Press, 1991

(汤帆)

paiduilun

排队论 (queueing theory) 研究服务系统中排队现象随机规律的一门学科。它的主要目标是研究如何以较少的投资获得最优服务质量的问题。每种服务系统都有三个基本组成:

(1) 输入过程 即顾客(人,信息或作业)到来的规则,通常考虑如下的模型:①定长输入,即顾客等间隔到达;②泊松流输入,即假定在 t 时间内到达 k 个顾客的概率 $P_k(t) = \frac{e^{-\lambda t} (\lambda t)^k}{k!}, k=0,1,\dots,\lambda$ 为单位时间内到达顾客的个数;③其他输入,如一般独立输入,埃尔朗输入,成批到达输入等。

(2) 排队规则 ①损失制,即顾客得不到服务就自动消失;②等待制,其中又分先到先服务、后到先服务、随机服务、优先权服务等方式;③混合制。

(3) 服务机构模型 单个或多个服务台;并联或串联服务台;单个服务或成批服务。服务时间又可分定长分布、指数分布、埃尔朗分布等。

D. G. Kendall 引入一个常用的关于随机服务系统分类的记号 $X/Y/n$,其中 X 为输入分布, Y 为服务分布, n 为服务台数目,并以 M 代表泊松输入或指数的服务分布, D 为定长分布, E_k 为埃尔朗分布, GI 为一般独立输入, G 为一般服务分布。

如果系统已运行了相当长的时间,系统状态不再受初始条件的影响,此时称系统处于稳态,对于 $M/M/n$ 系统稳态的一个著名公式是埃尔朗公式:顾客到达系统后,由于 n 个服务台被占用而遭受损失的概率为:

$$P_n = \frac{(\lambda/\mu)^n}{n!} / \sum_{j=0}^n \frac{(\lambda/\mu)^j}{j!}$$

式中 λ 为泊松流强度, μ 为指数服务分布的指数,由此可求得在给定损失率下服务台的最小数目。

另一著名的公式是辛钦公式:在 $M/G/1$ 系统中顾客的平均数:

$$Q = \rho + (\rho^2 + \lambda^2 \sigma^2) / 2(1 - \rho)$$

式中 $\rho = \lambda/\mu$ 称为服务强度, σ^2 是顾客服务时间的方差。由此可知,在 λ, ρ 不变时,减小 σ^2 (即使每个顾客服务时间的差异尽可能地小)可使 Q 下降。

排队论在实时系统,分时系统,计算机网络和存储器分配中都有广泛应用。下面举例说明排队论在操作系统设计中的应用。

(1) 批处理系统和前端处理系统 这里的顾客

就是每一个作业或每一条信息,假定它们是泊松输入(参数为 λ),CPU处理每个作业的时间服从指数分布(参数为 μ)。于是得到一个 $M/M/1$ 模型。以 P_n 表示稳态时系统中有 n 个作业的概率,则有: $P_n = \rho^n (1 - \rho)$,其中 $\rho = \lambda/\mu < 1$ 。①假定缓冲区容量为 k ,则系统因溢出而丢失的概率为 $P_r = \sum_{i=k+1}^{\infty} P_i = \rho^{k+1}$;②系统中平均等待的作业个数为 $\bar{Q} = \sum_{n=1}^{\infty} n P_n = \rho / (1 - \rho)$;③系统中作业的平均等待时间为 $w = \frac{1}{\mu - \lambda}$ 。

(2) 多用户分时系统模型 假设系统有一台主机CPU与 N 个终端构成,输入与服务规律同(1),则可求得在稳态时,系统中没有等待的作业(即全部完成)的概率为:

$$P_0 = \left(\sum_{i=0}^{\infty} \frac{N!}{(N-i)!} \rho^i \right)^{-1}$$

在交互式分时系统中,响应时间 R 是一个重要的技术指标,它包括用户从打入命令到收到系统响应的时间间隔,可求得平均响应时间 $\bar{R} = \frac{N}{\mu(1-P_0)} - \frac{1}{\lambda}$ 。假定 $\mu = 0.5s, \frac{1}{\lambda} = 15s$,根据上述公式可绘出 \bar{R} 与 N 的关系曲线,并且看出当终端配置超过20台后, \bar{R} 将迅速增大。

参考文献

1. 徐光辉. 随机服务系统. 北京: 科学出版社, 1980
2. 李永锡等. 计算机操作系统原理. 西安: 西北工业大学出版社, 1987 (金治明)

paixu suanfa

排序算法 (sorting algorithm) 数据处理中将文件中记录按键码的一定次序要求排列起来的算法。在讨论排序算法时,数据通常是指由若干记录组成的文件,每个记录包含一个或多个数据项,其中能够标志该记录的数据项称为键码。给定一文件的 n 个记录 $\{R_1, R_2, \dots, R_n\}$ 及其相应的键码集合 $\{K_1, K_2, \dots, K_n\}$,所谓排序就是将记录按键码递增次序排列起来。当待排序的文件能够同时装入计算机的主存中时,则相应的排序称为**内排序**;如果文件大到不能同时全部装入主存中而有一部分必须放在外存上时,则相应的排序称为**外排序**。当待排序的文件中包含有一些相同键码的记录时,如果经过排序后

这些相同键码的记录相对次序仍然保持不变,则相应的排序算法是稳定的,否则为不稳定的。如果排序算法设计成单处理机完成的,则此排序算法称为**串行(或顺序)排序算法**;如果排序算法设计成多处理机实现的,则称为**并行排序算法**。度量串行排序算法复杂度的标准是算法的运行时间和所占用的存储空间;度量并行排序算法复杂度的标准是算法的总运行时间和所需的处理器数。排序的应用很广,在科学计算和数据处理中,在数据库和知识库管理系统中,在系统软件和应用软件中以及在高级计算机体系结构中,都会直接或间接地遇到大量的排序问题。排序在计算机科学研究中占有相当的地位,人们已经发现,排序问题的研究方法和思路,算法的设计和分析技巧,对研究计算机诸多领域中其他问题的算法都颇值得借鉴。内排序的方法很多,最常用的有插入排序、选择排序、交换排序(包含快速排序、堆排序)、分配排序和归并排序等。外排序多采用多路归并方法。

插入排序的基本方法是:每次将一个待排序的记录 R_i ,按其键码 K_i 的大小插到以前已排序的文件中的适当位置,直到全部插入完为止。

选择排序的基本方法是:每次从待排序的记录中选出其键码最小的记录依次放在已排序的文件中,直到选完为止。

交换排序的基本方法是:两两比较待排序记录的键码,并交换那些不满足顺序要求的键码对,直到全部都满足为止。

分配排序又称为桶排序,它适合于记录有多个特征键码的文件排序。

归并排序的基本思想是:将一些已排序的子文件进行合并而得到一个完整的有序文件。归并时,只要比较各子文件的第一个记录的键码,其最小者就是全局最小者;取出它后,继续比较各子文件的第一个记录的键码,这样就得到全局的次最小者,如此下去,就可完成排序。

磁盘排序属于外排序。外排序方法与各种外存设备的特征有关。外存设备大体上可分为顺序存取(如磁带)和直接存取(如磁盘)两大类。

磁带排序也是外排序,其过程基本上与磁盘排序过程相同,即先对输入文件的各段进行内排序,生成初始顺串,再把它们写到带上;然后再把这些顺串进行反复地归并,直到剩下一个顺串为止。磁带排序时间主要取决于对带的读写。

并行排序是指利用多台处理机(并行机)进行

的排序,其目的主要是为了提高速度。并行排序算法虽然和前述的单处理机上的串行排序算法有不少相似之处,但不能认为它只是串行排序算法的简单推广和扩充。它的最大特点之一就是它和并行计算机的体系结构密切相关,不同体系结构导致不同加速和不同设计风格的并行排序算法。

参考文献

1. 克努特著. 计算机程序设计技巧(第三卷:排序和查找). 管纪文, 苏运霖译, 陆汝钤等校. 北京: 国防工业出版社, 1984
2. 陈国良. 并行算法: 排序和选择. 合肥: 中国科学技术大学出版社, 1990 (陈国良)

Peiteli wanglun

佩特里网论(Petri net theory) 一种以物理学为基础,用计算机科学语言提出的系统模型和理论。简称网论。适合于模拟和分析以资源(物质,数据,信息等)流动为特征的异步并发系统,其最终目标是要用一种兼容物理和计算机两个学科的语言和概念框架来形式描述制约通信进程的所有“自然法则”。

网论起源于1962年C. A. Petri的博士论文《用自动机通信》。20世纪60年代以研究佩特里网系统(简称网系统)个体的动态行为为主要内容,给出了计算所有可能状态的可覆盖树法,定义了S-不变量和T-不变量并给出了计算方法。网系统上的进程由出现网和从出现网到网系统的映射组成,既能正确描述系统中的顺序行为,也能准确表示并发行为。这一阶段的网论又称特殊网论。

70年代起是佩特里网论发展阶段,至今已有并发论、同步论、赋逻辑论和网拓扑等四个较成熟的分支。它们构成通用网论的基础。

网论尊重物理事件有其自身发生的条件和影响范围,网系统中没有控制流和中央控制的概念,除非它所描述的物理系统是顺序的或中央控制的。

中央控制需了解系统的全局状态。大系统的全局状态并非总是实时可知的。网论遵循局部确定原则,不依赖全局状态来确定事件之能否发生,因而适合于描述异步并发现象。

网论尊重时间是对变化的度量,认为时间就体现在变化之中,不使用实数作为时间模型。网系统中每个对象的活动轨迹是该对象的全序时间,不同对象交互作用之处则是两个全序时间“同时”之点。网系统中可以读同一个钟表的子系统,可以以此种

表之读数为其局部时间。

网系统以描述资源在系统中的流动为特征。物质流,数据流和信息流均是网系统中的“资源流”。

盛放资源的元素称为库所,改变资源种类和数量的元素称为变迁,连接库所和变迁,代表资源流动方向的是流关系。这三者依次用圆圈、方框和箭头表示时构成网状结构,称为佩特里网,简称网。各类资源的个数用相应圆圈中黑点的个数表示。这种黑点称为托肯。资源在库所中的分布称为网上的标识。网上加上初始标识就是佩特里网系统,简称网系统。标识决定变迁能否发生,变迁的发生则决定标识的变化。规定标识和变迁之间依存关系的法则称为变迁规则。网和标识给出网系统的静态结构,变迁规则给出网系统的动态行为。前者对应于程序系统中的语法,后者对应于语义。

网 由库所集 S , 变迁集 T 和流关系 F 构成的三元组 $N = (S, T; F)$ 是个网的条件是:

$$\begin{aligned} S \cup T &\neq \emptyset && \text{非空} \\ S \cap T &= \emptyset && \text{库所和变迁不同类} \\ \wedge F \subseteq S \times T \cup T \times S &&& \times \text{为笛卡儿积} \\ \wedge \text{dom}(F) \cup \text{cod}(F) &= S \cup T && \text{无孤立元素} \end{aligned}$$

其中 $\text{dom}(F) = \{x \mid \exists y: (x, y) \in F\}$, $\text{cod}(F) = \{y \mid \exists x: (x, y) \in F\}$ 分别是 F 的定义域和值域。

网系统 $\Sigma = (S, T; F, K, W, M_0)$ 为网系统的条件是: $(S, T; F)$ 为网, $K: S \rightarrow N \cup \{\infty\}$, $W: S \times T \cup T \times S \rightarrow N_0$ 及 $M_0: S \rightarrow N_0$, 其中 N_0 和 N 分别为含 0 和不含 0 的正整数集, 且 $W(x, y) = 0 \Leftrightarrow (x, y) \notin F$ 。

K 是库所上的容量函数, W 是箭头上的权函数, M_0 是初始标识。

图形表示 图 1 是网系统的图形表示, 其中 $S = \{s_0, s_1, s_2, \dots, s_9\}$, $T = \{t_0, t_1, t_2, \dots, t_7\}$, $F = \{(s_0, t_1), (s_1, t_1), (t_1, s_3), \dots, (s_7, t_7), (t_7, s_1)\}$,

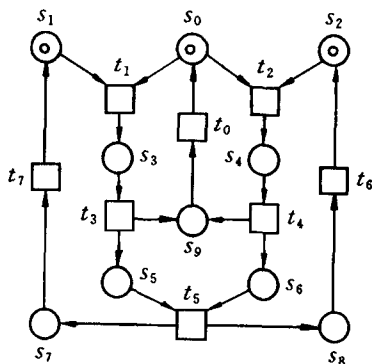


图 1 网系统的图形表示

$M(s_0) = M(s_1) = M(s_2) = 1$, 而对 $i \geq 3$, $M(s_i) = 0$ 。

变迁规则 设 $t \in T$ 为任一变迁, t 的前集 $\cdot t$ 和后集 $t \cdot$ 分别定义为: $\cdot t = \{s \mid (s, t) \in F\}$ 和 $t \cdot = \{s \mid (t, s) \in F\}$ 。令 $M: S \rightarrow N_0$ 为任一标识 (即 $\forall s \in S: M(s) \leq K(s)$), 则 t 在 M 有发生权的条件是 $\forall s \in S: W(s, t) \leq M(s) \leq K(s) - W(t, s)$, 即其前集库所中有足够的资源, 后集库所中有足够的容量。 t 在 M 有发生权记作 $M[t >]$ 。此时若 t 发生, 则将标识 M 改变为 M 的后继 M' , M' 的定义是, 对所有 $s \in S$, $M'(s) = M(s) - W(s, t) + W(t, s)$ 。 M' 为 M 的后继用 $M[t > M']$ 表示。

图 1 中 t_1 和 t_2 各自均能发生, 但不能同时发生, 因为它们共享的 s_0 类资源只有一个, 而它们各需要一个。网论中称 t_1 和 t_2 在 M_0 互相冲突, 冲突是网论揭示的基本现象之一。注意图 1 中没有明确给出容量函数 K 和权函数 W 。通常这表示容量均为 ∞ , 权均为 1。

基本现象 包括顺序、冲突、冲撞、并发和混感等。资源的产生和消耗是顺序关系, 共享资源的缺乏引起冲突, 容量不够导致冲撞, 互不依赖就是并发。混感是并发和冲突的混合引起的无法判断冲突是否有增减的现象。

并发是提高效率的手段。顺序是被迫的。冲突代表非确定性, 是系统环境作决策或行控制之处。冲撞是可以消除的, 只要把容量也当作资源直接用库所表示即可。混感是系统不完整的表现, 必须避免。

进程 把系统中变迁的发生和资源的流动如实记录下来, 就是一个进程。图 2 给出了图 1 中网系统的一个进程。进程中的网称为出现网。其特点是有无向环路且每个库所至多有一个出的和一个入的箭头。出现网上标出的 s_0, s_1, \dots 和 t_0, t_1, \dots 指明各元素与原系统中库所和变迁的对应。所以网论中的进程是个有序偶 (\mathcal{N}, ρ) , 其中 \mathcal{N} 是出现网, ρ 是从 \mathcal{N} 到原系统的映射。显然网论的进程准确地表现了原

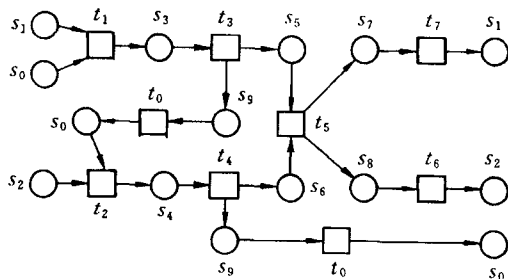


图 2 进程

系统中的顺序(如 t_1 和 t_3)和并发(如 t_6 和 t_7)。 t_5 所起的同步作用也很明显。

网系统是不需加以解释(给出库所和变迁的具体含义)即可分析动态行为的系统模型。

不变量 图 1 中的网系统是个循环系统,因为只要按一定顺序让变迁 t_0 发生两次,其他变迁各发生一次,它就会恢复到它的初始标识。依次以变迁 t_0, t_1, \dots, t_7 为序标的列向量 $(2, 1, 1, 1, 1, 1, 1, 1)^T$ 称为该系统的 T -不变量。 T -不变量也可能只涉及系统的部分变迁。另一方面,无论系统处于从 M_0 经变迁发生可以到达的什么标识,库所 s_0, s_3, s_4 和 s_9 中的托肯总数永远是 1。这就是系统的 S -不变量,对应的以 s_0, s_1, \dots, s_9 为序标的列向量是 $(1, 0, 0, 1, 1, 0, 0, 0, 0, 1)^T$ 。循环系统或循环子系统与 T -不变量对应,资源全部或部分守恒则对应着 S -不变量。这些不变量都是与网系统对应的线性方程组的解。

可达标识集 从 M_0 经变迁发生可以到达的所有标识 i 集合,可以用来回答系统是否有界,是否安全,是否有死锁及是否有死变迁等问题。网论给出了有效算法,可以构造有限网系统的可覆盖树。所有可达标识都被此树覆盖。

变迁序列 从 M_0 出发可以依次发生的变迁所成之序列称为变迁序列。所有变迁序列的集合代表着系统的行为,也是构造佩特里网语言的基础。佩特里网语言是将每个变迁映射到一个字符得到的。

高级网系统 不同类的资源可能起着类似的作用,不同的变迁也可能有相似之处。前者如图 1 中的 s_1 和 s_2, s_3 和 s_4 。后者如同一系统中的 t_1 和 t_2, t_3 和 t_4 。将类似的库所和变迁合并,用带记号的托肯(称为个性托肯)代替黑点托肯,就得到高级网系统。这种网系统节点少,便于分析。常用的有谓词/变迁系统和有色网系统。

其他网系统 常见的有随机网系统和时间网系统。

容量函数恒为 1 的网系统是信息系统。这时每个库所代表一位信息。也可将库所看成条件:或成真(有托肯)或不成真,这时对应的变迁称为事件。若像其他网系统一样只研究从初始状态出发的“未来”行为,就得到基本网系统。若还允许逆箭头而“上”,研究它过去的行为,则得到条件/事件系统,简称 C/E 系统。C/E 系统是通用网论的基础。通常用 B 和 E 代替 S 和 T 分别表示条件集和事件集,而且把 C/E 系统中正向和反向可以到达的标识称

为情态。

同步距离 任取 $E_1, E_2 \subseteq E$, 用 E_1 作前集, E_2 作后集,可以构造出一个 S -类元素。这个元素若不属于 B ,即不是系统中给出的条件元素,就可以用来观察 E_1 事件和 E_2 事件的动态关系。办法是把它当作容量为无穷的库所,而且假定其中有足够多的托肯。当 E_1 中事件正向发生一次或 E_2 中事件反向发生一次,它得到一个托肯;若 E_1 中事件反向发生一次或 E_2 中事件正向发生一次,它失去一个托肯。其中托肯总数最大值和最小值的差就是 E_1 和 E_2 之间的同步距离,记作 $\sigma(E_1, E_2)$ 。若最大值不存在, $\sigma(E_1, E_2) = \infty$, $\sigma(E_1, E_2)$ 满足距离公理。已证明,顺序事件的同步距离为 1, 并发或冲突事件的同步距离则不小于 2。同步距离是同步论的核心概念。

赋逻辑 若 $B_1, B_2 \subseteq B$ 为不相交的两个条件子集,则分别以 B_1, B_2 为前集和后集可以构造出一个 T -类元素。若这个 T -类元素在任何情态下均不能发生,就说它是死的。设 $B_1 = \{a_1, a_2, \dots, a_n\}, B_2 = \{b_1, b_2, \dots, b_m\}$, 则这个死的 T -类元素对应着命题 $a_1 \wedge a_2 \wedge \dots \wedge a_n \rightarrow b_1 \vee b_2 \vee \dots \vee b_m$ 在所有情态成真。这样,死的 T -类元素就可以用来对 C/E 系统的情态进行命题演算。已证明关于情态的任何有效命题均可用一个或几个这种死元素表示。死元素上的扩张规则和消解规则及反证法(当 $n=0=m$ 时,死元素代表矛盾)可以完成命题演算及命题证明。这些都可以代数化,用矩阵的初等变换来完成。

C/E 系统和模态逻辑和时态逻辑同样密切相关。

并发论 网论提出了 22 条并发公理,以刻画并发现象的根本特性。例如并发关系的不传递性及其传递核和传递闭包;并发的非平凡性,自返性,对称性,不可化简性,相干性, K -稠密性, N -稠密性及并发的连续性。

网拓扑 拓扑学中的开集公理来源于实数轴。由此导出的连续性隐含着全序、稠密及实数势,而且每个点都成闭集。实际应用中往往要偏序集,有限或可数无限集上的连续性和稠密性。将开集公理中有限个开集之交仍为开集放松为任意多个开集之交仍为开集,就得到网拓扑。这是沟通连续模型和离散模型的途径。

网系统允许层次模拟,从最细节的出现结构到最抽象的系统,都以网结构为基础。不同层次之间可以由在网拓扑之下的连续映射沟通。这就使网论

应用有了理论基础。佩特里网论已经或正在成功地应用于网络通信、软件工程、柔性制造、系统控制及 CAD 等领域。工业上的国际和国家标准《控制系统功能图表的绘制》即是以佩特里网为基础制定的。目前已有构造和分析网系统的计算机辅助工具上市。

参考文献

1. Brauer W. Net Theory and Applications. Springer-Verlag, 1979, LNCS Vol. 84
2. Brauer W, Reisig W and Rozenberg G. Petri Nets: Applications and Relationships to Other Models of Concurrency. Springer-Verlag, 1986, LNCS Vol. 254, 255
3. 袁崇义. Petri 网. 南京: 东南大学出版社, 1989 (袁崇义)

peizhi guanli gongju

配置管理工具 (configuration management tool)

用于支持软件配置管理活动的工具软件。通常而言,配置管理要求对标识出的软件产品及其中间产品进行有效的存取、控制和审计等,由此需要解决针对软件资源的访问、存取等控制要求配置管理工具,在此基础上还可以对包括并行开发、远程开发、系统构造等软件生产的相关问题提供支持。配置管理工具是软件工具中的基本支持工具,能够完成配置管理中的主要非智力性工作,提供版本管理(含压缩、浏览、访问、合并等)、人员权限、并发与变更控制等多方面的自动支持功能。常见的配置管理工具有以独立工具形式出现,也有以开发环境成分存在的。

配置管理工具的发展可追溯到版本存储系统,早期工具(如 UNIX 环境下的 SCCS 系统)主要解决各种版本存储和访问问题,一般通过版本树的结构进行管理。随着软件系统的复杂性日益提高和计算机网络技术的发展,配置管理工具逐步将并行开发、变更管理作为基本功能,在可视化版本空间管理、编译环境集成和团队开发等方面提供了越来越强的功能。全面地看,软件配置管理系统通常包括 8 个方面的主要功能:①存储 是指对中间软件产品的存储和维护。一般通过增殖存储等方式解决版本存储的冗余问题,提高存储效率;并通过形象化的版本树提供访问机制。②结构 是指对软件系统各构件及其结构的关系描述与管理,同样包括这种关系演化的管理。③构件 是指利用系统结构定义建立实际

的系统,包括构建和部署。④审计 提供对软件配置管理活动的记录和分析功能。⑤统计 提供对各种统计信息的查询和报告。⑥控制 一般通过人员权限、定义变更生存周期的方式,实现对配置项的变更。⑦过程 是指维护与管理软件配置管理活动相关的软件过程,一般可看作是软件过程管理技术在配置管理中的应用。⑧团队 是指对多个开发人员协同进行软件开发的支持,主要通过加锁和分配策略对工作区管理、并行开发管理和远程开发管理提供支持。

配置管理工具可以从两个角度来分类:①按版本对象的粒度,可以分为基于文件的系统、基于目录的系统和基于构件与体系结构的系统。早期的系统以文件为版本管理对象,所有处理对象皆为独立的文件;后期发展了基于目录的系统,能够将目录作为一个文件集合进行管理,记录目录的版本;现在的主要系统均以构件和体系结构为管理对象,可管理不同层次对象的版本,并适应基于构件的软件开发方法。②按用户直接处理的对象类型,可以分为面向版本的系统和面向变更的系统。前者用户以版本树为管理对象,处理各类版本信息;后者以变更为主,版本只是某一变更的相关属性。

国内外有许多独立的商用配置管理工具,如 Rational 公司的 Clearcase、CA 公司的 CCC-Harvest、Luccent 公司的 Sablime、北京大学的 JBCM 等,也有 CVS 等免费软件系统。同时,在各类开发环境中也都可见配置管理功能的存在,如 Microsoft 公司 Visual Studio 中的 Visual Source Safe、Borland 公司开发环境产品中的 PVCS 等。

参考文献

- Burrows C, George W, Dart S. Configuration Management. Ovum Ltd., 1996 (谢冰)

penquan moxing

喷泉模型 (fountain model)

一种体现软件创建所固有的迭代和无“间隙”特征的软件开发模型。如图 1 所示。

这一模型刻画了软件开发活动需要多次重复。例如,在编码之前(设计之后),再进行分析和设计,期间,添加有关功能,使系统得以演化。同时,该模型还表明活动之间没有明显“间隙”,例如分析活动和设计活动之间没有明显的界线。

喷泉模型主要用于描述面向对象开发过程。由于对象概念的引入,表达分析、设计、实现等活动只

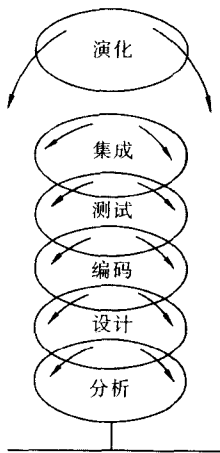


图 1 喷泉模型

用对象和关系,因而,可以容易实现活动的重复以及活动之间无明显“间隙”。并且,使其开发自然地包括复用。
(王立福)

pichuli

批处理 (batch processing) 把多个作业积累成批,处理过程中用户不能再行干预的处理方式。

采用批处理方式处理作业时,用户把要处理的作业,包括程序、数据及作业说明书一起交操作员或通过终端输入到计算机,然后,由系统按作业说明书来控制执行。

提供批处理功能的操作系统称**批处理操作系统**,它根据预定的策略按某种组合和次序选择待处理的作业去执行。例如,选择计算型和输入输出型用户作业搭配运行,可使系统各类资源均衡使用,提高系统效率。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989
(郑宇华)

pianshang xitong

片上系统 (system on a chip, SOC) 具有完整系统功能的一种复杂芯片。至少包括处理器、局部存储器、接口和专用处理部件。它源于 1998 年前后,是超大规模集成电路之后的新发展阶段的标志,支持它的技术是深亚微米 CMOS 工艺和可将上亿量级(10^8)的器件集成到单一芯片的工艺。

SOC 尚未有权威的定义。其中文名称也未统一,常见的有芯片上系统、系统级芯片、系统集成芯

片等。一种集成电路芯片如果具备了下述特征,可称为 SOC。这些特征是: ①具有实现复杂系统功能的 VLSI; ②采用深亚微米工艺技术; ③使用一个或数个嵌入式 CPU、MPU 或 DSP; ④具有从外部对芯片进行编程的功能; ⑤配套有嵌入式软件和操作系统或实时操作系统; ⑥主要采用第三方的 IP(知识产权)核进行设计。为了缩短进入市场时间, SOC 采用已经被验证过的 IP 核进行设计,因此,可重用的成功设计越多,其设计过程会越快。

SOC 的内涵随技术的进步而改变,由于它比一般的集成电路具有更高的集成度,因而更适合于专用。SOC 在开始时的应用领域主要是个人通信类产品和信息家电产品。随着技术的进步,有可能将模拟电路、数字模拟混合电路以及无线通信用的射频模块、随机存储器等都集成在同一个 SOC 中。

对 SOC 一类的复杂芯片,测试是一个难题。因此,在设计各个阶段都要考虑可测试性问题,这样可测试性设计技术(DFT)便得以发展。SOC 的设计过程包括体系结构和算法设计、IP 选择、DFT 设计、系统集成、验证及物理综合等。

为了进一步缩短进入市场的时间,降低 SOC 的开发风险、开发成本和小批量生产成本,一些半导体生产厂商推出了多种可编程片上系统产品,即 SOPC 产品。SOPC 在芯片中预先定制了一些 CPU 的硬核、SRAM、定点乘法器、双端口存储器和接口电路,并提供几十万至几百万可编程逻辑门。此外,还可以提供各类 IP 核。这类芯片兼具定制逻辑与可编程逻辑的优点,提高了 SOPC 的性能,缩短了产品的开发风险和开发周期,是 SOC 的主要发展趋势。

信息技术正在逐步改变着人们的日常生活。在这个改变过程中, SOC 做出了重大贡献,也取得了飞速发展。SOC 的年增长率超过 30%,其市场潜力是巨大的,许多半导体公司将 2/3 的研究开发资源投入 SOC 领域。
(时万春 韩承德)

pian weifen fangcheng shuzhi jiefa

偏微分方程数值解法 (numerical solution of partial differential equation) 根据偏微分方程特点,研究在计算机上如何采用有效的数值方法求解偏微分方程的方法和理论。用离散的网格覆盖求解区域,按某种方式将连续形式的偏微分方程离散成以网格点(结点)上的函数值为未知量的代数方程组,求解方程组得到结点上未知函数的值。这是大型数字计算机在科学技术领域的重要应用,涉

及科学与工程计算的许多问题,范围相当广泛。

偏微分方程是联系自变量与未知函数及其偏导数的等式。一个重要的类型是两个变元的二阶线性偏微分方程,其一般形式是

$$Lu = Au_{xx} + 2Bu_{xy} + Cu_{yy} + Du_x + Eu_y + Fu + G = 0 \quad (1)$$

其中 L 是微分算子, u 是未知函数, A, B, C, D, E, F 和 G 依赖于 x 和 y , u_x 表示 u 对 x 的偏导数。方程 (1) 在点 (x, y) 是椭圆型、双曲线型或抛物型,按判别式 $B^2 - AC$ 在该点为负、为正或为零而定。典型的例子有

$$\Delta u = f(x, y) \quad (\text{椭圆型}) \quad (2)$$

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2} \quad (\text{双曲线型}) \quad (3)$$

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} \quad (\text{抛物型}) \quad (4)$$

其中 $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$, 称为拉普拉斯算子, $f(x, y)$ 是已知函数, a 为常数。以上三个方程也依次分别称为泊松方程 ($f(x, y) = 0$ 时称为拉普拉斯方程)、波动方程和热传导方程。

偏微分方程的两类重要定解问题是边值问题和初值问题,以方程 (2) 和 (4) 为例有下面的定义。

求 $u(x, y)$ 在有界区域 Ω 内满足方程 (2), 在 Ω 的边界上满足 $u = \varphi(x, y)$ (φ 是已知函数), 称这样的定解问题为狄利克雷问题或第一边值问题; 若在边界上 $\partial u / \partial n = \varphi$ (n 为边界的外法线方向), 称这样的边值问题为诺依曼问题或第二边值问题。

求 $u(t, x)$ 在区域 $\{|x| < \infty, t > 0\}$ 内满足方程 (4), $t = 0$ 时 $u(0, x) = u_0(x)$ (u_0 是已知函数), 称这样的定解问题为初值问题或柯西问题。

如果一个定解问题的解存在、惟一且连续地依赖于定解条件, 则此定解问题称为适定的。定解问题只有在很特殊的情况下才能用解析的方法求解, 一般情况下采用数值方法, 如差分法、有限元法等求解。

差分法

差分法是把定解区域剖分为网格, 在网格结点上用差商代替微商, 将微分方程化为包含有限个未知数的差分方程组。差分法直观、简单易行, 能普遍地用于各种类型的微分方程和任意形状的区域。但因为它包含巨大的运算量, 只有在电子计算机问世之后才得到广泛的应用和发展。

边值问题差分法 求解椭圆型方程边值问题的

一类数值方法。作为例子考虑泊松方程的狄利克雷问题:

$$\left. \begin{aligned} \Delta u &= f, \text{ 在 } \Omega \text{ 内} \\ u|_{\partial\Omega} &= \varphi, \text{ 在 } \partial\Omega \text{ 上} \end{aligned} \right\} \quad (5)$$

其中 $\partial\Omega$ 表示区域 Ω 的边界。

区域 Ω 的网格剖分, 一种最简单又常用的方法是以平行于坐标轴的两族直线为网格线, 作正方形网格剖分得网格 Ω_h (图 1)。结点坐标为 $x_i = ih$, $y_j = jh$, h 称为步长 (两相邻平行线间的距离), i, j 为整数。记 $u_{ij} = u(x_i, y_j)$, 当 (x_i, y_j) 是内点时, 用差商 $(u_{i+1,j} + u_{i-1,j} - 2u_{ij}) / h^2$ 代替 $\partial^2 u / \partial x^2$, 用差商 $(u_{i,j+1} + u_{i,j-1} - 2u_{ij}) / h^2$ 代替 $\partial^2 u / \partial y^2$, 得到微分方程的差分方程 (差分格式)

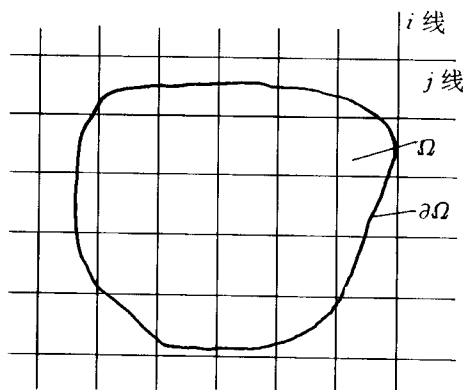


图 1

$$(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij}) / h^2 = f_{ij} \quad (6)$$

格式 (6) 常称为五点差分格式, 可简记为: $\Delta_h u_h = f_h$, 其中 Δ_h 是对应于微分算子 Δ 的差分算子, u_h 和 f_h 是定义在 Ω_h 上的网函数: $u_h(ih, jh) = u(ih, jh)$, $f_h(ih, jh) = f(ih, jh)$ 。

对于靠近边界的结点, 如果结点正好落在边界上, 则取相应的边界值。一般情况下, 用偏心差商列出不等距差分方程, 也可用其他方法 (如插值法) 作边界处理。

对于一般的微分方程 $Lu = f$, 类似的方法用差分算子 L_h 代替微分算子 L 逼近微分方程, 得到差分方程 $L_h u_h = f_h$ 。当 u 充分光滑时, 称 $R_h(u) = [L_h u]_{ij} - [Lu]_{ij}$ 为差分方程的截断误差, 其中 h 的最低幂次称为截断误差的阶。当网格步长趋于零时, 差分格式的截断误差也趋于零, 则称差分格式是微分方程的相容逼近。相容性说明网格步长越小,

差分方程与微分方程越接近,截断误差的阶反映了逼近的精度。

差分方程 $L_h u_h = f_h$ (含边界条件) 实际上已是线性或非线性方程组,常用的求解方法是直接法和迭代法。针对差分方程的特点,现已开发了一些求解偏微分方程的软件包,如 ELLPACK,它由输入、划分网格、离散化、方程求解及输出等模块组成。用迭代法求解椭圆差分方程组时,可应用 ITPACK 软件包中的有关模块。用直接法时可用 LINPACK 或稀疏矩阵软件包 Yale 中有关模块。

初值问题差分法 它是求解双曲型方程和抛物型方程这类含时间 t 的偏微分方程的初值问题或初边值混合问题的一类数值方法。以一阶双曲型方程的初值问题为例

$$\left. \begin{aligned} \frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} &= 0, \quad |x| < \infty, t > 0 \\ u(0, x) &= u_0(x), \quad |x| < \infty \end{aligned} \right\} \quad (7)$$

取时间步长 τ 和空间步长 h , 作 xt 平面的网格剖分(图2)。记 $u_j^n = u(n\tau, jh)$, 用差商 $(u_{j+1}^n - u_j^n) / \tau$ 代替 $\partial u / \partial t$, 若用一阶中心差商 $(u_{j+1}^n - u_{j-1}^n) / (2h)$ 代替 $\partial u / \partial x$, 则得式(7)的一个差分格式

$$u_j^{n+1} = u_j^n - \frac{1}{2} r (u_{j+1}^n - u_{j-1}^n) \quad (8)$$

当 $a > 0$ 时,可用向后差商 $(u_j^n - u_{j-1}^n) / h$ 代替 $\partial u / \partial x$, 得格式

$$u_j^{n+1} = u_j^n - r(u_j^n - u_{j-1}^n) \quad (9)$$

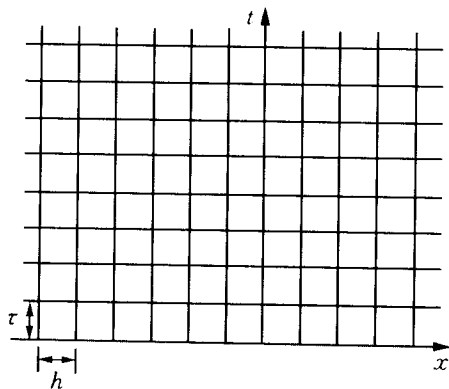


图 2

如果 $a < 0$, 则用向前差商 $(u_{j+1}^n - u_j^n) / h$ 代替 $\partial u / \partial x$, 得格式

$$u_j^{n+1} = u_j^n - r(u_{j+1}^n - u_j^n) \quad (10)$$

其中 $r = a\tau/h$ 。

格式(8)的截断误差是 $O(h^2 + \tau)$, 格式(9)和格式(10)的截断误差是 $O(h + \tau)$, 它们都满足相容性。根据差分格式以及初值条件 $u_j^0 = u_0(jh)$, 可依次求出 u_j^1, u_j^2, \dots 。

差分格式的稳定性 对于偏微分方程的初值问题,差分格式的稳定性对于实际数值计算至关重要。若稳定性的研究主要针对初始条件,则称为关于初值的稳定性。它要求初值 u_j^0 小的改变引起 u_j^n 的改变也小。对于常系数线性偏微分方程的差分格式的稳定性,诺依曼运用傅里叶分析作了系统的研究。把差分方程的解表示成谐波的叠加。考虑其中一个谐波

$$u_j^n = G^n(\sigma, \tau) e^{i j \sigma h} \quad (11)$$

的增长情况。其中 σ 为实数, i 为虚数单位, $G(\sigma, \tau)$ 称为增长因子。若对一切谐波(11)的振幅一致有界,即存在常数 $M > 0$, 对一切适合 $0 \leq n\tau \leq T$ (T 为某一取定的常数)的 n 和充分小的 τ 都有 $|G^n| \leq M$, 则此差分格式是稳定的。

显式格式与隐式格式 考虑抛物型方程的初值问题

$$\left. \begin{aligned} \frac{\partial u}{\partial t} &= a^2 \frac{\partial^2 u}{\partial x^2}, \quad |x| < \infty, t > 0 \\ u(0, x) &= u_0(x), \quad |x| < \infty \end{aligned} \right\} \quad (12)$$

记 $\delta^2 u_j = u_{j+1} - 2u_j + u_{j-1}$, 用差商 $(1/h^2)\delta^2 u_j^n$ 代替 $\partial^2 u / \partial x^2$, $\partial u / \partial t$ 仍用 $(u_j^{n+1} - u_j^n) / \tau$ 代替, 可得差分格式

$$u_j^{n+1} - u_j^n = r \delta^2 u_j^n \quad (13)$$

这里 $r = a^2 \tau / h^2$ 。若初值是以 1 为周期的函数, 且假设 $u_0(0) = u_0(1) = 0$, 设 $N = 1/h$, 则还应加上边界条件: $u_0^{n+1} = u_N^{n+1} = 0$ 。

若用差商 $(1/h^2)\delta^2 u_j^{n+1}$ 代替 $\partial^2 u / \partial x^2$ 时, 则得差分格式

$$u_j^{n+1} - u_j^n = r \delta^2 u_j^{n+1} \quad j = 1, 2, \dots, N-1$$

$$u_0^{n+1} = u_N^{n+1} = 0$$

(14)

比较这两个格式,前者可以由 n 层结点上的 u_j^n 直接计算 u_j^{n+1} , 这称为显式格式。后者是含未知数 u_j^{n+1} ($j = 1, 2, \dots, N-1$) 的方程组, 不能直接计算 u_j^{n+1} , 必须解方程组, 故称为隐式格式。

拉克斯等价定理 对于线性偏微分方程组的适定的初值问题, 一个与之相容的线性差分格式收敛的充分必要条件是这个格式是稳定的。

多重网格法

该法是求解差分方程组(网格方程)的一类数

值方法。

多重网格法将求解区域用不同步长进行剖分,得到多个层次的由粗到细逐步加密的网格,把细层网格上求解问题转化为较粗层(直到最粗层)网格上求解。

多重网格迭代是以二重网格迭代为例,考虑边值问题(5)。用两种步长 h 和 $H(h < H)$ 剖分得网格 Ω_h 和 Ω_H ,并有差分算子 L_h 和 L_H , Ω_h 上的离散方程是

$$L_h u_h = f_h \quad (15)$$

设 u_h^j 是式(15)的一近似解,用 $e_h^j = u_h - u_h^j$ 表示 u_h^j 与离散方程精确解 u_h 的误差, $d_h^j = f_h - L_h u_h^j$ 是 u_h^j 的残量,从而有误差方程

$$L_h e_h^j = d_h^j \quad (16)$$

由式(16)的解和近似解 u_h^j 可得式(15)的解: $u_h = u_h^j + e_h^j$ 。

线性转换算子 I_h^H 与 I_H^h 分别称为限制算子和延拓算子, $I_h^H: G(\Omega_h) \rightarrow G(\Omega_H)$, $I_H^h: G(\Omega_H) \rightarrow G(\Omega_h)$ 。其中 $G(\Omega_h)$ 表示 Ω_h 上的网函数空间。又设给定了解 Ω_h 上的离散方程(15)的一个松弛法: $\bar{u}_h^j = S(u_h^j, L_h f_h)$,并用 S^k 表示执行 k 次 S 松弛,则有二重迭代过程如下:

(1) 对 u_h^j 用松弛法 S 作 k_1 次松弛得 $\bar{u}_h^j = S^{k_1}(u_h^j, L_h f_h)$ 。

(2) 粗网格校正得 \bar{u}_h^j : ① 计算残量 $\bar{d}_H^j = f_h - L_h \bar{u}_h^j$; ② 限制残量 $\bar{d}_H^j = I_h^H \bar{d}_h^j$; ③ 在 Ω_H 上解残量方程 $L_H \bar{e}_H^j = \bar{d}_H^j$ 得 \bar{e}_H^j ; ④ 延拓校正得 $\bar{e}_h^j = I_H^h \bar{e}_H^j$; ⑤ 计算校正近似值 $\bar{u}_h^j = \bar{u}_h^j + \bar{e}_h^j$ 。

(3) 对 \bar{u}_h^j 用松弛法 S 作 k_2 次松弛得 $u_h^{j+1} = S^{k_2}(\bar{u}_h^j, L_h f_h)$ 。

以上过程反复进行直到某个 u_h^{j+1} 满足精度要求为止。(1)和(3)中的 k_1 和 k_2 的选取依具体问题而定,尚无统一的选取原则。由上述迭代过程可见,多重网格法是一个细网格上松弛和粗网格上校正来回交替迭代的过程,其优点是计算量较小而收敛速度快。

区域分裂法

并行求解偏微分方程特别是椭圆型方程的一类数值计算方法。

区域分裂法将定解问题的求解区域分成 m 个子区域 $\Omega_i: \Omega = \bigcup_{i=1}^m \Omega_i$ 。相应地将原问题分解成 m

个较小的子问题,每个子问题独立求解,同时还须建立子问题之间的某些相互联系与通信,以保证它们的总体效果收敛到原问题的解。这种思想早在1870年就由德国数学家H. A. Schwarz提出,并用于解非凸平面区域上拉普拉斯方程的狄利克雷问题。1890年E. Picard用它解一类非线性椭圆型方程,称之为施瓦茨交替法。但直到20世纪70年代末期,它才随着并行计算机的问世而发展成为解偏微分方程的一类并行数值方法。此后产生了子区域重叠和不重叠的区域分裂法,对称区域分裂法等多种类型,并由此产生了空间分裂法等

施瓦茨交替法 考虑平面拉普拉斯方程狄利克雷问题

$$\left. \begin{aligned} \Delta u &= 0, & \text{在 } \Omega \text{ 内} \\ u|_{\partial\Omega} &= \varphi \end{aligned} \right\} \quad (17)$$

其中边界 $\partial\Omega$ 由不相切的两部分 \widehat{ACB} 和 \widehat{ADB} 组成(图3)。将 Ω 分裂成两个子区域 Ω_1 和 Ω_2 ($\Omega_1 \cap \Omega_2 \neq \emptyset$),它们的边界分别是 \widehat{ACBFA} 和 \widehat{AEBDA} ,其中 \widehat{AEB} 和 \widehat{AFB} 称为拟边界。于是原问题分解成两个子问题:

$$\left. \begin{aligned} \Delta u &= 0, & \text{在 } \Omega_1 \text{ 内} \\ u|_{\widehat{ACB}} &= \varphi, & u|_{\widehat{AFB}} &= g_1 \end{aligned} \right\} \quad (18)$$

$$\left. \begin{aligned} \Delta u &= 0, & \text{在 } \Omega_2 \text{ 内} \\ u|_{\widehat{ADB}} &= \varphi \\ u|_{\widehat{AEB}} &= g_2 \end{aligned} \right\} \quad (19)$$

其中 g_1 和 g_2 分别是 \widehat{AFB} 和 \widehat{AEB} 上的连续函数,且 $g_1(A) = \varphi(A) = g_2(A)$, $g_1(B) = \varphi(B) = g_2(B)$,它们由交替迭代过程确定并随迭代步变化。

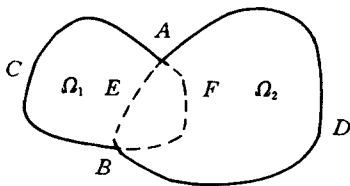


图 3

解问题(17)的施瓦茨交替法如下:

(1) 给问题一个初始猜测 u^0 ;

(2) 令 $g_1|_{\widehat{AFB}} = u^0|_{\widehat{AFB}}$,求解问题(18)得

$$\bar{u}^1, \text{延拓 } \bar{u}^1 \text{ 到 } \Omega \text{ 得 } u^1 = \begin{cases} \bar{u}^1, & \text{在 } \Omega_1 \text{ 内} \\ u^0, & \text{在 } \Omega \setminus \Omega_1 \text{ 内;} \end{cases}$$

(3) 令 $g_2|_{\overline{AB}} = u^1|_{\overline{AB}}$, 解问题 (20) 得 u^2 , 同法延拓得 u^2 。

重复执行 (2) 和 (3) 两步, 即交替地解两个子问题, 得到函数序列 $u^0, u^1, \dots, u^{2n+1}, u^{2n+2}, \dots$ 。此序列的极限即为问题 (17) 的解, 这由下面的定理所保证。

施瓦茨定理 若问题 (17) 有惟一解且它在 Ω 内二次连续可微, 则序列 $\{u^i\}$ 在 Ω 的任一内子区域上一致收敛于此解。

在交替过程中还可以引进松弛因子 ω , 若记 (2) 和 (3) 的解为 \hat{u}^{2n+1} 和 \hat{u}^{2n+2} , 并分别加入下列松弛过程

$$u^{2n+1} = u^{2n} + \omega(\hat{u}^{2n+1} - u^{2n}) \quad (20)$$

$$u^{2n+2} = u^{2n+1} + \omega(\hat{u}^{2n+2} - u^{2n+1}) \quad (21)$$

其中 $0 < \omega < 2$ 。这样做成的序列 $\{u^i\}$ 也收敛到问题 (17) 的解, 且若 ω 选得合适, 则它较原序列收敛得更快。这就是带松弛因子的施瓦茨交替法。

混乱松弛与并行算法 松弛步 (2) 和 (3) 是有先后顺序的, 它的直接推广是将 Ω 分裂为 m 个子区域的系统松弛法。它按区域 Ω_j 的编号顺序循环地松弛得 $u^{m+j}(t = 0, 1, 2, \dots; j = 1, 2, \dots, m)$, 即第 $mt+j$ 次松弛总是在第 j 个子区域上进行。混乱松弛则是松弛次序允许千变万化。记 $i(j)$ 表示第 i 次松弛是在第 j 个子区域上进行, 则混乱松弛法可描述为: 给一初始猜测 u^0 , 在 Ω_k 上松弛得 $u^{1(k)}$, 在 Ω_l 上松弛得 $u^{2(l)}, \dots$, 照此下去得到序列 $\{u^{i(j)}\} (i = 1, 2, \dots; k, l, j = 1, 2, \dots, m)$ 。这样做成的序列并不能保证收敛, 因为不加限制的标号序列 $\{i(j)\}$ 可能导致有的子区域无限次被松弛, 而有的子区域只松弛有限次。为此将序列 $\{i(j)\}$ 按子区域的编号分成 m 个子序列: $\{i_1(1)\}, \{i_2(2)\}, \dots, \{i_j(j)\}, \dots, \{i_m(m)\}$ 。 $\{i_j(j)\}$ 是对第 j 个子区域松弛的那些序号的序列, 并规定当 $i \rightarrow \infty$ 时, 有 $i_j(j) \rightarrow \infty (j = 1, 2, \dots, m)$ 。有了这样的规定, 则 $\{u^{i(j)}\}$ 收敛到问题 (17) 的解 (在施瓦茨定理的条件下)。

混乱松弛的计算机实现便是并行算法。设有 m 台处理机, 每台处理机负责一个子区域上的松弛过程。这些过程在运行时除了拟边界上的近似解的最新信息相互交换之外, 各自独立并行执行, 彼此不需要等待, 故是一类异步并行算法。当然为了保证必要的精度要求与逻辑的正确性, 还应当有某些作为判断依据的控制信息参与过程之间的

通信。

参考文献

1. 李荣华, 冯果忱. 微分方程数值解法. 北京: 高等教育出版社, 1980
2. 哈克布思. W. 多重网格法. 北京: 科学出版社, 1988
3. 康立山, 孙乐林, 陈毓屏. 解数学物理问题的异步并行算法. 北京: 科学出版社, 1985

(李元香 康立山)

pinfen duolu fuyong

频分多路复用 (frequency division multiplexing, FDM) 采用分频技术把传输线路的可用带宽分割成多个频段, 并将各个频段分配给各个输入信号的多路复用技术。

频分多路复用 (FDM) 首先用来把许多路电话合并在一起, 在一根电缆上传输。每一路电话需要 300 ~ 3 000 Hz 的带宽, 仅占用一根铜线的可用总带宽的一部分。双绞线电缆的可用带宽是 100 kHz, 因此, 在同一根双绞线电缆上可采用频分复用技术多路传输多达 24 路电话。频分多路复用工作过程如图 1 所示。

第一步: 滤出 12 部电话机中每一部电话机的话音频率。公认的话音频率大约为 300 Hz ~ 3.3 kHz。另外, 从这一步中看到的滤波器叫做低通滤波器, 其作用是阻止 4 kHz 以上的频率进入系统 (允许稍低于 300 Hz 或稍高于 3.3 kHz 的任何话音频率进入系统)。

第二步: 对第一步中滤出的话音信号进行调制。一种常用的方法是把话音信号调制成一个高频模拟信号。每一个输入话音信号均被调制到频谱中不同的位置上。

第三步: 这时 FDM 系统又要使用滤波器。这些滤波器叫做带通滤波器, 其任务是滤掉包含话音信息频率之外的所有频率。

第四步: 到这一步就已经获得了 12 个信号, 每个信号都代表一路电话所包含的信息。最后一步是接收作为 12 个子信道的这些信息, 并将它们作为一个信号在一条传输线路上同时传输出去。

接收端的多路复接器包含有只接收某些频带的滤波器, 具体说, 这些滤波器只接收能在发送端通过滤波器的那些频带。

虽然这一信号中含有多种不同的频带, 似乎非常杂乱无章, 但每个频带所包含的信息均可由 FDM

系统多路复接器一端的滤波器送到合适的地点。

FDM 只适用于模拟信号,对于数字信号的应

用,需要用调制器将信号转换成模拟信号,然后再利用 FDM 技术进行多路传输。

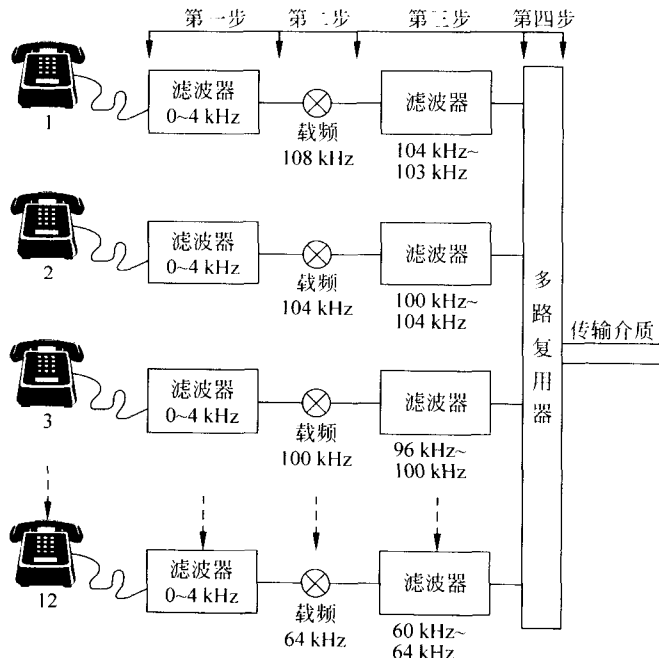


图 1 频分多路复用技术

参考文献

1. 胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002
2. Sheldon T. LAN Times Encyclopedia of Networking. McGraw-Hill Inc., 1994 (胡道元)

pingfang bijin

平方逼近 (approximation in quadratic norm)

$$\text{用 } \|f-g\|_2 = \left[\int_a^b [f(x)-g(x)]^2 \omega(x) dx \right]^{\frac{1}{2}}$$

来度量函数 f 与 g 的距离的逼近(参见数值逼近)。

就平方逼近来说,被逼近函数 f 可以属于比连续函数类 $C[a,b]$ 更广的函数类,即所有使 $\int_a^b |f(x)|^2 \times \omega(x) dx$ 存在的 f 之集合,记作 $L_\omega^2[a,b]$ 。定义 $L_\omega^2[a,b]$ 中两个函数 f 与 g 的内积为

$$(f,g) = \int_a^b f(x)g(x)\omega(x)dx$$

当 $(f,g) = 0$ 时,称 f 与 g 正交。设 $\varphi_0, \varphi_1, \dots, \varphi_n$ 为 $L_\omega^2[a,b]$ 中一组线性无关的函数,它们的所有线性组合所构成的函数集合记作 φ , 则对于每个

$f \in L_\omega^2[a,b]$, 在 φ 中的最佳平方逼近

$$S_n(x) = \sum_{k=0}^n C_k \varphi_k(x)$$

存在且惟一,系数 C_k 通过求解 $n+1$ 阶线性代数方程组

$$\sum_{i=0}^n (\varphi_i, \varphi_k) C_i = (f, \varphi_k), \quad k = 0, 1, \dots, n$$

而得

$$C_k = \frac{G(\varphi_0, \dots, \varphi_{k-1}, f, \varphi_{k+1}, \dots, \varphi_n)}{G(\varphi_0, \varphi_1, \dots, \varphi_n)}, \quad k = 0, 1, \dots, n$$

其中

$$G(\varphi_0, \varphi_1, \dots, \varphi_n) = \begin{vmatrix} (\varphi_0, \varphi_0) & (\varphi_0, \varphi_1) & \dots & (\varphi_0, \varphi_n) \\ (\varphi_1, \varphi_0) & (\varphi_1, \varphi_1) & \dots & (\varphi_1, \varphi_n) \\ \dots & \dots & \dots & \dots \\ (\varphi_n, \varphi_0) & (\varphi_n, \varphi_1) & \dots & (\varphi_n, \varphi_n) \end{vmatrix}$$

称为格拉姆行列式。特别地,当 φ_k 为正交函数系时, C_k 可简化为

$$C_k = \frac{(f, \varphi_k)}{(\varphi_k, \varphi_k)} \quad (k = 0, 1, \dots, n)$$

并称为 f 的广义傅里叶系数, 而 $S_n(x) = \sum_{k=0}^n C_k \varphi_k(x)$ 称为 f 按正交函数系 $\{\varphi_0, \varphi_1, \dots, \varphi_n\}$ 的级数展开式。且有最小偏差

$$\|S_n(x) - f(x)\|_2 = (\|f\|_2 - (S_n f))^{1/2}.$$

当 $\varphi_k(x)$ 为代数多项式时就得到最佳平方逼近多项式, 若 $\varphi_k(x)$ 为 $[-1, 1]$ 上的勒让德多项式 $P_k(x)$, 则 f 在 $[-1, 1]$ 上的最佳平方逼近多项式为

$$S_n(x) = \sum_{k=0}^n \frac{2k+1}{2} (f, P_k) P_k(x)$$

这里 $P_0 \equiv 1, P_k(x) = \frac{1}{2^k k!} \frac{d^k}{dx^k} [(x^2 - 1)^k], k = 1, 2, \dots, n$ 为勒让德多项式。

若 $\varphi_k(x)$ 为 $[-1, 1]$ 上的切比雪夫多项式 $T_k(x)$, 则 f 在 $[-1, 1]$ 上的最佳平方逼近多项式为

$$S_n(x) = \sum_{k=0}^n C_k T_k(x)$$

$$\text{其中 } C_0 = \frac{1}{\pi} \int_{-1}^1 \frac{f(x)}{\sqrt{1-x^2}} dx,$$

$$C_k = \frac{2}{\pi} \int_{-1}^1 \frac{T_k(x) f(x)}{\sqrt{1-x^2}} dx, \quad k = 1, 2, \dots, n$$

$$T_k(x) = \cos(k \arccos x).$$

它可作为 f 在 $[-1, 1]$ 上的近似最佳一致逼近多项式。

平方逼近的计算过程非常方便, 故很有实用价值。

参考文献

Davis P J. Interpolation Approximation. Blaisdell, Waltham, Mass, 1963 (沈毅)

pingjun guzhang jiang shijian

平均故障间隔时间 (mean time between failures, MTBF) 在计算机系统运行的一个较长时间段内, 两次相邻故障间隔的平均时间。

平均故障间隔时间是衡量计算机系统可靠性的一个重要指标, 它的值越大表示计算机系统的可靠性越高, 可连续正常运行的时间越长。它的同义词有平均无故障时间和平均无差错时间。其计算公式为

$$MTBF = \frac{t}{N(t) + 1}$$

式中 t ——计算机系统运行的一段较长的时间;

$N(t)$ ——系统在 $[0, t]$ 时间段内的故障次数。

在计算机发展初期, 由于所用元器件的寿命短, 可靠性差, 因而系统的平均故障间隔时间短, 只有几小时或十几小时。后来, 由于集成电路等元器件技术的发展, 再加上各种容错技术的应用, 现代计算机系统的可靠性已大大提高, 其平均故障间隔时间可达成千上万小时。

在计算机系统设计中进行可靠性预测时, 可用组成系统的各种成分的失效率来计算 MTBF:

$$MTBF = \frac{1}{\lambda}$$

式中 λ ——系统的失效率。

λ 是各分系统、设备或器件的失效率的总和:

$$\lambda = \sum_{i=1}^n \lambda_i$$

式中 λ_i ——分系统、设备或器件的固有失效率。

每种器件在材料和工艺水平相同的情况下都有相对固定的失效率。这个失效率可以在器件的实际使用中得到, 也可以通过抽样做加速寿命实验获得。可按下式计算平均失效率 $\bar{\lambda}_i$:

$$\bar{\lambda}_i = \frac{n}{NT}$$

式中 n ——失效元件个数;

N ——参加运行元件个数;

T ——运行时间。

如果是通过加速寿命实验获得, 还要除以加速因子, 求出在正常工作环境下的失效率:

$$\bar{\lambda}_i = \frac{n}{NT\sigma}$$

式中 σ ——加速因子, 可根据实验条件或查已有的加速因子表获得。

通过失效率计算出来的 MTBF 只是对系统可靠性的估算, 不包括偶发性故障在内。这种估算方法也广泛应用于其他电子设备中。

参考文献

猪濑·博编著. 计算机系统的高可靠性技术. 尤国峻, 肖俊远译. 高铭学, 肖兴权校. 北京: 国防工业出版社, 1985 (应秋丽 苏振泽)

pingmiantu

平面图 (planar graph) 能够在平面上画出, 且边不在非顶点处相交的**无向图**。在平面上画出的边不在非顶点处相交的平面图 G 的图形称为 G 的平面表示。在平面图 G 的一个平面表示中, 以 G 的

边为边界的连通区域称为 G 的该平面表示的面。一个平面图可以有多个不同的平面表示,但其任何平面表示的面的数目均一样。这可以用著名的欧拉公式进行计算;若 n 阶平面图 G 有 m 条边和 k 个分支,则 G 的一个平面表示的面数 $f = m - n + k + 1$ 。在无向图 G 的任一条边 e 上插入一个度为 2 的顶点,将 e 一分为二,或者从 G 中删去一个度为 2 的顶点 v ,将与 v 关联的两条边合二为一,可得到图 G' ,此时称 G 与 G' 同胚。波兰数学家 Kuratowski 给出了判断一个图是否为平面图的准则,即著名的库拉托夫斯基定理:无向图 G 是平面图当且仅当 G 没有同胚于图 1 或图 2 的子图(参见图论)。

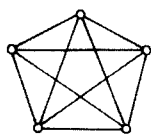


图 1

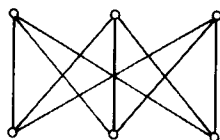


图 2

给定平面图 G 的一个平面表示,在其每个面内找一顶点,若两个面有 m 条公共边,则用 m 条线连接这两个面内取定的顶点,并使其分别与 m 条公共边相交。由这些顶点和连线组成了边不在非顶点处相交的图形。称以该图形为其平面表示的平面图为 G 的对偶图。(张强)

poufen qumian

剖分曲面 (subdivision surface) 基于一组拓扑规则和几何规则对初始多边形网格递归地进行细化所得到的网格序列的极限。又称细分曲面或子分曲面。网格细化所遵循的规则称为剖分模式。其中拓扑规则用于确定新网格顶点的插入方法及连接关系,而几何规则用来计算新网格顶点的位置。剖分曲面思想简洁,适于任意拓扑网格且具有多分辨率结构。

剖分方法最早可以追溯到 1950 年左右 G. de Rham 用以描述光滑曲线的角切削思想。1974 年 G. Chaikin 给出了基于这一思想的一个曲线生成方法——Chaikin 算法。E. Catmull 和 J. Clark 于 1978 年从双三次 B 样条的递推性质导出任意拓扑网格上的剖分曲面模式。同一年提出的 Doo-Sabin 剖分曲面则是双二次 B 样条的推广。

剖分理论的一个重要内容是极限曲面的连续性分析。Doo 和 Sabin 通过构造剖分矩阵并基于离散

傅里叶变换进行矩阵特征分析,对剖分模式的收敛性和相应剖分曲面的连续性进行了探讨。这一方法后来成为研究剖分曲面性质的核心技术。不过,严格的连续性理论直到 1995 年后才由 U. Reif 和 D. Zorin 等人建立。与此同时,剖分曲面在图形学领域得到了广泛应用。尽管如此,奇异点处二阶连续曲面的构造迄今尚无令人满意的结果。动态剖分曲面构造及其连续性分析框架尚未建立。

虽然已提出了大量剖分曲面模式,剖分曲面的性质及不同模式之间的关系值得进一步研究。为了满足 CAD 应用的需求,在曲面属性的快速计算、灵活的数据结构、有效的形状控制、多分辨率编辑、高精度的剖分曲面求交和裁剪及基于剖分曲面的逆向工程等方面仍有许多问题亟需解决。

参考文献

Denis Zorin, Peter Schröder. Subdivision for Modeling and Animation. In: Proceedings of ACM SIGGRAPH'2000 Course Notes #23, 23 ~ 28 July, 2000, New Orleans, Louisiana (彭群生)

pubu moxing

瀑布模型 (waterfall model) 一种软件开发模型,该模型将软件生存周期的各项活动规定为依固定顺序联接的若干阶段工作,形如瀑布流水,最终得到软件产品。

瀑布模型可追溯到 50 年代末期,当时人们已感到必须先确认“做什么”,才能编制程序将其实现,即使是比较简单的小型问题也不例外。最简单的两级瀑布模型如图 1 所示。

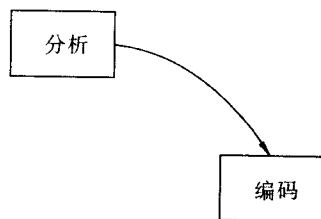


图 1 两级瀑布模型

对于较大项目,问题更加复杂,两级模型已不能满足软件开发的实际需要。一个更精确的软件开发步骤可按需要解决问题的顺序依次为:做什么—如何做—制作—检测—使用。于是一个反映软件过程的基本框架如图 2 所示。该图表明,首先应给出软件的目标,确定要做什么,然后要决定如何达到这一

目标,给出策略、方法和步骤;继而加以实施,制作出所要的软件;经过适当的检测,判定符合初始目标以后,方可投入运行和使用。可以说这是瀑布模型的雏型。

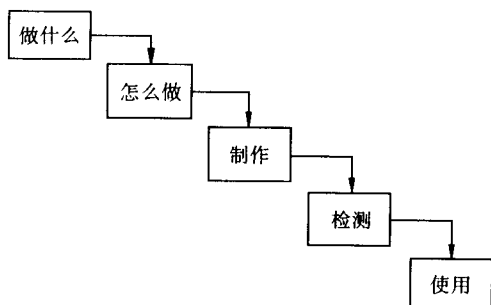


图2 瀑布模型雏型

1970年W. Royce首先将这一模型精确化,提出了具有多个开发阶段的瀑布模型,如图3所示(不包括几个向上的虚线箭头)。这一模型规定了开发各阶段的活动为:提出系统需求、提出软件需求、需

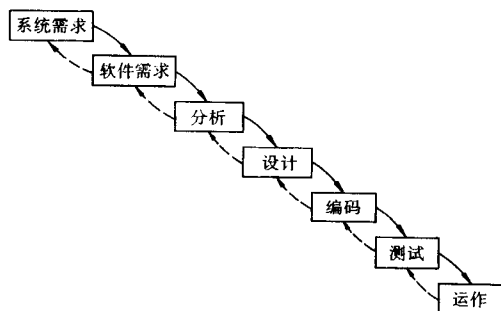


图3 瀑布模型

求分析、设计、编码、测试和运作。并且还规定了自上而下相互衔接的固定顺序。因而构成了熟知的瀑布模型。然而实践表明,各开发阶段间的关系并非完全是自上而下的线性图式。实际情况是,每个开发阶段均应具有以下特征:

- (1) 从上一阶段接受本阶段工作的对象,作为输入;
- (2) 对上述输入实施本阶段的活动;
- (3) 给出本阶段的工作成果,作为输出传入下一阶段;
- (4) 对本阶段工作进行评审,若本阶段工作得到确认,则继续下一阶段工作,否则返回前一阶段,甚至更前阶段。

为表达向前阶段的反馈,在图3中增加了虚线表示的箭头,从而构成了具有反馈回路的瀑布模型。

瀑布模型有着不同形式的变种。比如,另一常见的具有反馈回路的瀑布模型包含的7个阶段是:可行性研究、需求分析和规约、设计和规约、编码和单元测试、集成测试和系统测试、交付、维护。不同形式瀑布模型的变种彼此并无本质差别,选择何种形式由软件项目特性及开发机构决定。

许多采用瀑布模型的开发机构为有效地组织实施,制定了软件开发规范或开发标准。其中明确规定了各个开发阶段应交付的产品,这就为严格控制软件开发项目的进程,最终按时交付产品以及保证软件产品的质量创造了有利条件。

参考文献

Carlo G, et al. Fundamentals of Software Engineering. Prentice Hall, 1991

(郑人杰)

Q

qiye liucheng chongzu

企业流程重组 (business process reengineering, BPR) 对企业业务流程作根本性的思考和彻底重建的过程和技术。其目的是在成本(C)、质量(Q)、服务(S)和速度(T)等方面取得显著的改善,使得企业能最大限度地适应以顾客、竞争和变革为特征的现代企业经营环境。企业流程重组(BPR)对“显著性”有具体的要求,如将生产周期缩短70%,成本降低40%,顾客满意度、产品质量和总收入均提高40%等。

企业流程重组理论是在传统的劳动分工理论基础上发展起来的。进入20世纪80年代以来,随着市场竞争的不断加剧和新技术革命的迅速发展,三种因素(顾客、竞争和变革)使得企业面临十分严峻的形势,传统的经营管理模式日益显露出弊端。1990年,美国著名企业管理学者、麻省理工学院 Michael Hammer 提出了 BPR 理论。1993年,Michael Hammer 与咨询专家 James Champy 合著并出版了《企业重构——经营管理革命的宣言书》,指出 BPR 遵循的指导思想为:①按照市场需求决定企业的业务流程;②由业务流程来决定企业的组织结构;③用灵活应变、基于项目组形式的组织结构替换传统的组织结构;④采用扁平分布化的组织结构,纵向压缩组织,横向加大管理幅度;⑤权力下放,尽可能缩小企业中央管理部门的职能,将其限制在核心部分;⑥增加业务部门的权限;⑦将供应商和顾客也包括到企业内部业务过程中;⑧克服官僚主义作风,加速业务流程和部门之间的通信。在上述思想指导下,企业流程重组包括人的重组、技术的重组、组织结构的重组和企业文化的重组等多个方面。

企业流程重组(BPR)的实施要依靠工业工程技术、运筹学方法、管理科学、社会人文科学和现代高科技,并且涉及到企业的人、经营过程、技术、组织结构和企业文化等各个方面。从信息技术角度而言,BPR 涉及的相关技术包括计算机网络与通信技术、数据库技术、决策支持系统技术,过程模型化与仿真技术、快速原型系统开发技术和项目管理工程工具等。

自企业流程重组(BPR)理论诞生以来,其思想的先进性与变革的彻底性引起了管理学界和企业界极大关注,许多知名公司纷纷实施 BPR,在企业市场占有率、投资回报率及顾客服务的速度与质量等综合指标上已取得了巨大的成效。

参考文献

1. Hammer M, Champy J. Reengineering the Corporation: A Manifesto for Business Revolution. Harper Business, 1993
 2. Barber M I, et al. Scoping Study on Business Process Reengineering Towards Successful 21 Application. Int. J. Prod. Res. 1998, 36(3): 576 ~ 601
 3. Hammer M, Stanton S A. The Reengineering Revolution. New York: Campus Verlag Frankfurt, 1995
- (仲伟俊)

qiye ziyuan gui Hua

企业资源规划 (enterprise resource planning, ERP) 一种基于系统化管理思想和信息技术为企业决策层及员工提供管理与决策的平台和技术。从管理角度看,它形成了一整套的企业管理系统体系标准,其实质是在“物料需求计划”(MRP)和“制造资源计划”(MRP II)基础上进一步发展而成的面向供应链的管理思想。从软件角度看,它是综合应用客户机-服务器体系结构、关系数据库、面向对象技术、图形用户界面、第四代语言(4GL)、网络通信等信息技术,体现企业资源规划管理思想的大型综合性软件。从管理系统角度看,它是整合企业管理理念、业务流程、基础数据、计算机硬件和计算机软件于一体的企业资源管理系统。

企业资源规划(ERP)是在 MRP 和 MRP II 基础上发展起来的。20 世纪 60 年代—70 年代开始发展的物流需求计划(MRP)是基于物料库存计划管理的生产管理系统,其核心目标为确定物料的需求量,并制定相关的生产计划,以降低库存和生产管理成本。80 年代出现了制造资源计划(MRP II)系统。MRP II 的基本思想是:基于企业经营目标制定生产计划,围绕物料转化组织制造资源,实现按需

要、按时进行生产。随着市场竞争的加剧,企业竞争空间与范围的扩大和市场与客户需求变化的复杂化,MRP II 很难适应新的市场环境,更不能满足供应链管理的要求。因此,90 年代初美国信息分析咨询公司 Gartner Group 在总结 MRP II 软件发展趋势时,借助于供应链管理的思想,提出了企业资源计划 ERP 的概念。与 MRP II 相比,ERP 除了包括和加强了 MRP II 各种功能之外,更加面向全球市场,功能更为强大,所管理的企业资源更多,支持混合式生产方式,管理覆盖面更宽,并涉及企业供应链管理,从企业全局角度制定经营与生产计划,是制造企业的综合集成经营系统。ERP 采用的计算机技术也更加先进,形成了集成化的企业管理软件系统。

企业资源规划所包含的管理思想涉及物流学、价值链、业务流程重组、敏捷制造、准时生产、精益生产、全面质量管理(TQM)、约束理论(TOC)、MRP II、供应链管理(SCM)、客户关系管理(CRM)等。它充分利用信息技术,将这些先进的管理思想集成起来,使其核心技术具有如下特点:一是具有完整、强大、开放的功能模块;二是具有数据采集和自动处理功能;三是实现智能化的商务应用层与数据库和显示层的连接和集成;四是具有先进、合理的企业业务流程模型。

ERP 系统在相关企业中已经得到了广泛应用。目前在大公司中应用最多的 ERP 系统是 SAP 软件。ERP 技术的发展趋势为:①更加面向市场,包含基于知识的市场预测、订单处理与生产调度、基于约束的调度功能等,具有更强的企业优化能力;②与制造执行系统(MES)、车间层操作控制系统(SFC)更紧密结合,形成实时化的 ERP/MES/SFC 系统;③ERP 的供应链管理功能将更强,并进一步面向全球化市场环境,强调供应商、制造商与分销商间的新的伙伴关系;④ERP 将更好地支持多种不同的制造方式,包括离散制造方式和流程制造方式;⑤ERP 的工作流管理功能进一步增强,通过工作流实现企业的人员、财务、制造与分销间的集成,并能支持企业经营过程的重组;⑥ERP 纳入产品数据管理(PDM)功能,增加对设计数据与过程的管理,并进一步加强生产管理系统与计算机辅助设计(CAD)、计算机辅助制造(CAM)系统的集成;⑦在计算机技术方面,ERP 将以浏览器-服务器分布式结构、多数据库集成与数据仓库、面向对象方法和 Internet、外联网等为核心技术,开发 ERP 应用系统。

参考文献

1. Brwone J. et al. Production Management Systems—A CIM Perspective. Addison-Wesley, 1988
2. Mcilvaine B. User Are Behind Many of the Big Changes in ERP, Managing Automation. 1996, pp. 26 ~ 28
3. 陈荣秋. 生产计划与控制——概念、方法与系统. 武汉: 华中理工大学出版社, 1995 (仲伟俊)

qifa shi sousuo

启发式搜索 (heuristic search) 一种利用与待解问题有关的信息(称为启发信息),对搜索路径的走向给予一定约束或选择的搜索方法。

搜索方法的目标是要在与问题有关的状态空间或图表示中,根据已知的初始状态(起始节点)、目标状态(满足目标状态描述的节点)以及从一种状态(节点)转换到另一种状态(节点)所允许的操作或算符,寻找一条从初始状态达到目标状态的途径。绝大多数问题求解技术最终都归结为状态空间或图的搜索问题。

一般说来,不同的问题求解类型需要不同的搜索策略。根据问题求解的任务和问题本身所存在的解的情况,问题求解可分为三种类型。一是问题只有惟一解或多个解,但它们均处于同等地位,不涉及寻找最优解。这类问题要求搜索方法尽可能地减少搜索次数并保证完全性,即问题存在解的话,搜索一定能成功并找到问题的解。定理证明所面临的就是这类问题。二是问题有多个解,问题求解的目的是寻求其最优解。在问题的规模不太大,复杂性不甚高的情况下,这是可以做到的,但对大多数这类问题来说,需利用某些启发信息以提高搜索效率。 A^* 和 AO^* 等启发式搜索算法所要解决的就是这一类问题。第三类与第二类相似,但问题是 NP 难解的(参见 NP 完全性理论)。在现实的存储资源和时间条件下很难或根本得不到最优解。同时,对于诸如推销员旅行问题等具体应用,令人满意的解也并非一定要最优解。因而在求解这类问题时可以放弃最优解而研究各种更加实用有效的启发式搜索方法。

20 世纪 50 年代末期, A. Newell, J. C. Shaw 和 H. A. Simon 开始研究启发式搜索。60 年代中期以后,随着计算机,尤其是人工智能应用领域的不断扩大, NP 难解性问题又长期得不到解决,因而启发式搜索的研究越来越引起人们的重视与兴趣,并且取

得了一批引人瞩目的成果。如 J. Doran 和 D. Michie 以及 N. J. Nilsson 的利用搜索估价函数引导搜索的方法, P. E. Hart, Nilsson 和 B. Raphael 的 A^* 算法, 与或图上的启发式搜索 AO^* 算法以及各种博弈树搜索等。

启发式搜索的最大特点就是在搜索过程中使用与问题有关的启发信息来缩减搜索量, 其一般过程如下:

步骤 1 建立只含有初始节点 S 的搜索图 G , 把 S 放入名为 OPEN 的未扩展节点表中;

步骤 2 建立扩展节点表 CLOSED, CLOSED 初始为空表;

步骤 3 若 OPEN 为空表, 则搜索失败并退出;

步骤 4 把 OPEN 表上的第一个节点 $node$ 移入 CLOSED 表;

步骤 5 若 $node$ 为目标节点, 则搜索成功并退出。问题的解就是图 G 中从 $node$ 开始, 沿着指针到达 S 的一条路径(节点的指针在步骤 7 建立);

步骤 6 对节点 $node$ 进行扩展, 生成 $node$ 的未在其祖先节点中出现过的后继节点集合 M , 将 M 的成员添入图 G ;

步骤 7 对 M 中的每一个未在 OPEN 表和 CLOSED 表中出现过的节点设置指针指向 $node$, 并把这些节点放入 OPEN 表。对 M 中已在两表中的成员, 确定是否需要更改其指向 $node$ 的指针方向。对 M 中已在 CLOSED 表中的成员确定是否需要更改图 G 中通向它的每个后裔节点的指针方向;

步骤 8 按某种策略和规则或某种估价值重新排列 OPEN 表中的节点顺序;

步骤 9 返回到步骤 3。

上述过程中的步骤 8 对 OPEN 表上的节点进行排序, 以便选出一个当前“最好”的节点在步骤 4 进行扩展。排序的策略和方法决定了搜索的性质和效率。

如果对 OPEN 表中的节点进行排序时不利用任何与问题有关的信息而取任意顺序或仅依据某种特定的简单策略和方式, 则得到的过程就是无信息搜索或称为盲目搜索。典型的无信息搜索有宽度优先搜索和深度优先搜索等。一般说来, 无信息搜索在搜索过程中要生成过多的节点, 以至对许多现实问题而言很难或根本不能完成搜索任务。因而人们设法利用与问题有关的各种启发信息对 OPEN 表中的节点进行排序, 从而使得搜索有可能沿着最有希望的路径区段进行。

要实现对节点的排序, 关键是要给出某种测度, 用以估算候选节点向目标节点逼近的“希望”程度。这种测度称为搜索估价函数。

估价函数通常记为 f , 它提供了对节点的评价方法, 能够确定有可能位于通向目标节点的最佳路径上的节点。可以根据问题从不同角度出发定义估价函数, 例如节点处在最佳路径上的概率; 节点与目标节点之间的距离测度或差异性测度等。估价函数选取的合适程度取决于所掌握的与问题有关的信息。

估价函数确定以后就可以根据节点的估价函数值在前述算法中的步骤 8 对节点重新排序。下面是在上述算法基础上给出的一个典型的利用估价函数的启发式搜索算法, 其中假定所选估价函数 f 的函数值随节点的希望程度增大而减小。

步骤 1 把节点 S 放入 OPEN 表, 计算 $f(S)$;

步骤 2 如果 OPEN 表为空, 则失败退出, 无解;

步骤 3 从 OPEN 表中选择一个 f 值最小的节点 $node_i$ 。如果 f 值最小的节点有多个, 且其中一个为目标节点时, 选该节点为 $node_i$, 否则选其中任一节点作为 $node_i$;

步骤 4 把 $node_i$ 移出 OPEN 表, 放入 CLOSED 表;

步骤 5 如果 $node_i$ 是目标节点, 则成功退出, 求得一个解;

步骤 6 扩展 $node_i$, 生成其全部后继节点。对于 $node_i$ 的每个后继节点 $node_j$, 进行以下工作:

(1) 计算 $f(node_j)$;

(2) 如果 $node_j$ 不在 OPEN 表和 CLOSED 表中, 则利用其估价函数 f 的值把它填入 OPEN 表并从 $node_j$ 设一指向 $node_i$ 的指针, 以便找到目标节点后回溯解答路径;

(3) 如果 $node_j$ 已在 OPEN 表或 CLOSED 表中, 则比较刚计算出的 $f(node_j)$ 和 $node_j$ 已在表中的 f 值, 若新的 f 值小, 则以新值取代旧值; 从 $node_j$ 指向 $node_i$, 而不是指向它的父节点; 若 $node_j$ 在 CLOSED 表中, 则把它移回 OPEN 表;

步骤 7 转到步骤 2。

可以看出宽度优先和深度优先搜索等都是上述搜索算法的特例。比如选择 $f(node_i)$ 为 $node_i$ 的深度即可得到宽度优先搜索。

估价函数的选择对启发式搜索的性能和结果有决定性的作用。搜索方法的优劣通常可以用渗透率和有效分支系数来衡量。渗透率定义为

$$P = \frac{L}{T}$$

式中, L 是所求得的到达目标节点的路径长度;

T 是搜索期间所生成的节点总数。

有效分支系数与 L 和 T 则有如下关系:

$$\frac{B(B^L - 1)}{B - 1} = T$$

B 值反映了搜索走向在目标节点方向上的集中程度。

启发式搜索可能是不完全的,它有可能得不到存在的最优解。从本质上说,这是牺牲完全性来换取高效率 and 满意解。启发式方法极大地提高了搜索效率,使得大量实际问题可以得到令人满意的解决,从而在许多领域,特别是在人工智能应用中取得了良好的效果,成为人工智能的主要技术之一。近年来人们在更为深刻和更为广泛的意义上提出了启发式程序设计思想,从而把启发式搜索方法的研究提高到一个新水平。启发式方法的研究将继续对计算机科学和人工智能产生重大影响。

参考文献

1. 傅京孙,蔡自兴,徐光祐等. 人工智能及其应

用. 北京: 清华大学出版社, 1988

2. Mercadal P. Dictionary of Artificial Intelligence. New York: Van Nostrand Reinhold, 1990

3. Shapiro S C. Encyclopedia of Artificial Intelligence, 2nd ed. New York: John Wiley Sons Inc., 1992 (赵沁平)

qian zhao wei yitaiwang

千兆位以太网 (Gigabit Ethernet) 在以太网和快速以太网基础上发展起来的数据传输速率能达到 1 000 Mb/s, 同时又能达到和以太网相同的传输距离的一种高速局域网。千兆位以太网仍然采用以太网的带碰撞检测的载波监听多址访问-冲突检测 (CSMA/CD) 访问控制方式和帧格式, 但将数据传输速率从以太网的 10 Mb/s 提高到 1 000 Mb/s。

1995 年 11 月美国电气和电子工程师学会 (IEEE) 802.3 委员会组建了千兆位以太网技术标准研究组, 并于 1998 年和 1999 年完成了 IEEE 802.3z 和 IEEE 802.3ab 两个标准。千兆位以太网的基准 (参考) 模型如图 1 所示。

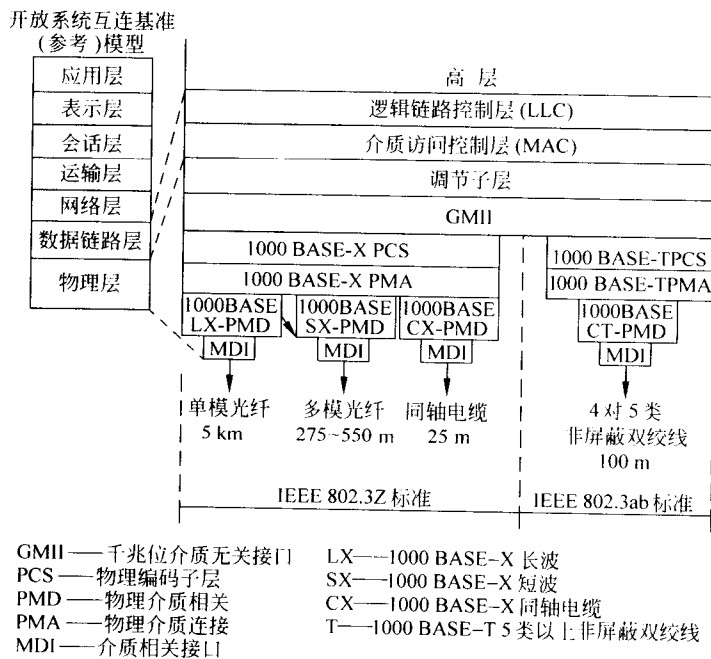


图 1 千兆位以太网基准(参考)模型

图 1 中 IEEE 802.3z 又称 1000 BASE-X, 是千兆位以太网的光纤接口标准和短距离线缆 (同轴电

缆) 接口标准。其中, 1000 BASE-LX 支持单模光纤和 150 Ω 的平衡同轴电缆, 光缆长度可达 5 km, 同轴

电缆长度只限于 25 m 以内;1000 BASE - SX 支持多模光纤,光缆长度在 275 ~ 550 m 之间;IEEE 802.3ab 又称 1000 BASE - T,是非屏蔽双绞线标准,使用 4 对 5 类非屏蔽双绞线(UTP),每对双绞线传输速率为 250 Mb/s,4 对汇聚成 1 000 Mb/s,双绞线电缆最大长度为 100 m。

千兆位以太网必须遵守以太网的通信规范,它仍然采用 CSMA/CD 介质访问控制方式,支持半双工或全双工操作(参见串行传输)。由于千兆位以太网的传输速度太快,很小的帧在传输时会使缆线接收的一端无法检测到冲突,所以采用了载波扩展和帧突发来改善其性能。载波扩展是在小于 512 字节的帧中,应用扩展符补齐。帧突发是当站有一批分组发送时,只对第一个分组用载波扩展补足扩展符,之后的分组一个接一个发送而不再补扩展符。载波扩展延长最小帧的传输时间从而延长了碰撞域。但它也影响了一些短帧的传输效率。帧突发则弥补了传输短帧效率低的影响。千兆位以太网使用标准的全双工通信,可以使数据传输速率从 1 Gb/s 提高到 2 Gb/s。

铜线的千兆位以太网用于在短距离内服务器和客户机连至交换机;光纤的千兆位以太网可用于中心结点之间的连接。千兆位以太网可用于高速服务器之间的连接、建筑物内的高速主干网、内部交换机的高速链路以及高速工作组网络。

千兆位以太网不仅满足了应用对网络速率和频宽的要求,而且较好地解决了和传统 10 Mb/s、100 Mb/s 以太网的兼容与升级,成为局域网主干网的主流技术。

参考文献

1. David Canningham. Gigabit Ethernet Networking. Macmillan Technical Publishing, 1999

2. 胡道元. 计算机局域网(第三版). 北京:清华大学出版社
(方起兴 相士俊)

qianduan chuliqu

前端处理器(front-end processor) 连接宿主机与若干个远程终端的一种通信控制设备。它是一台具有特殊能力的微型计算机,用以代替计算机中硬连线的通信控制器来对远程终端进行控制。

除了控制接收和发送所有在宿主机和终端之间传输的数据以外,前端处理器还可以完成下列各种功能:

(1) 数据和格式的转换 把一种或多种传输进

来的数据代码和格式转换成宿主机的数据代码或格式。

(2) 探询或选择 探询指前端处理器每次只请求一个终端发送数据,而选择则是前端处理器请求一个或多个终端接收数据。具有探询或选择功能的前端处理器可以确定某一终端是否已准备好发送或接收数据。

(3) 报文的装配或拆卸 对所有以不同的数据速率和按同步或异步格式传输的数据进行装配或拆卸,以保证宿主机收到的只是完整的报文。

(4) 差错控制 对传输差错进行检测并尽可能地校正。

(5) 故障弱化 指在宿主机系统的主要部件已经失效的情况下,该系统的另一些部件(例如终端)仍能保持继续工作的一种前端处理器的能力。

(6) 排队 把传输进来的报文按传输的次序排好以供宿主机处理。

(7) 报文交换 如果某个前端处理器能供不止一台宿主机使用,那么就可以用它来进行不同宿主机之间的报文交换。

(8) 直接响应 对于简单的查询,前端处理器可以直接响应而不必与宿主机打交道。

(过介望)

qianhou duanyan fangfa

前后断言方法(pre-and post-assertion method) 在语句前后分别加上前提条件(即前断言)和结果断言(即后断言),用程序设计逻辑证明程序正确性的方法。前提条件是该语句可执行的充分条件。结果断言指出语句执行中止后应具有的性质。前提条件又称前置断言,结果断言也称后置断言、后置条件等。

该证明方法源于 R. Floyd,他在 1967 年提出在流程图的连线上加上断言使得程序到达该处时断言为真。后经 C. A. R. Hoare 发展成型,成为程序设计逻辑的主要部分。

前后断言的主要形式有两种:

(1) $Q\{S\}R$ 刻画语句 S 的部分正确性。意指:若前提条件 Q 满足,且 S 执行终止,则 S 终止时结果断言 R 成立。

(2) $\{Q\}S\{R\}$ 刻画语句 S 的完全正确性。意指:若前提条件 Q 满足,则 S 执行终止,且 S 终止时结果断言 R 成立。

将 $Q\{S\}R$ 或 $\{Q\}S\{R\}$ 的整体作为一个谓词公

式,若该公式永真,则 S 相对于前提条件 Q 和结果断言 R 分别是部分正确或完全正确的。为证明复合语句 $Q \{ S_1; S_2 \} R$ 的部分正确性,可在 $S_1; S_2$ 之间插入断言 Q' 使得 $Q \{ S_1 \} Q'$ 和 $Q' \{ S_2 \} R$ 都永真。也可以插入多个断言 Q_1, Q_2, Q_3 等,使它们之间是蕴涵关系,用以反映证明思路。对其他语句构造可采用类似的方法。对于循环语句,除给出前后断言外,还须在循环中加入循环不变式,要求每次循环到达该处时不变式都成立。用前后断言可表示语句的公理语义。利用这些语义规则可帮助我们在程序正确性证明过程中确定该插入何种断言,以达到引导证明的目的(参见最弱前置条件方法)。前后断言方法,特别是结合最弱前置条件方法现已广泛用于规约语言及其支撑系统,如 VDM, Larch 等。

参考文献

1. Hoare C A R. An Axiomatic Basis for Computer Programming. CACM, 1969, 12(10): 576 ~ 580
2. Gries D. The Science of Programming. New York: Springer-Verlag, 1981
3. Dijkstra E W. A Discipline of Programming. Englewood Cliffs: Prentice Hall, 1976 (伊波)

qianrushishi caozuo xitong

嵌入式操作系统 (embedded operating systems) 为嵌入式电子设备提供的现代操作系统。嵌入的计算机及其操作系统与这些设备的其他部件密切地结合成一体。嵌入式电子设备泛指内部嵌有计算机的各种灵巧电子设备,这些电子设备的应用范围涉及信息采集、信息交流、通信娱乐等多种应用领域。嵌入式操作系统是嵌入在这些设备内部的计算机操作系统,它为设备实现各种灵活功能提供信息处理系统平台。从技术发展来看,一旦诸如移动电话、掌上电脑、媒体视听等设备的内部嵌入了计算机,就为设备的多样服务功能提供了坚实的技术基础。

嵌入式操作系统的主要特点是,它需要满足多种多样嵌入式设备的功能需求和满足设备应用环境的需求,主要包括:①尽量节约设备的电池耗电,提供能源管理功能;②应用中有不同档次的实时性要求,特别是满足言语、视频影像等信息服务的及时性要求;③高可靠性要求,要防止信息丢失、泄漏、恶意破坏等;④操作系统的易移植性要求,满足在多种硬件环境下安装和配置的需要。

目前常见的嵌入式操作系统有 Windows CE、VxWorks 和从 Linux 发展出来的嵌入式 Linux;具有专门用途的掌上电脑的 Plam OS 以及用于移动电话的 Symbian 等。

为了提高系统的易移植性,嵌入式操作系统通常采用硬件抽象层(HAL)和板支持组件(BSP)的底层结构设计。

参考文献

1. 陈向群,杨芙清. 操作系统教程. 北京: 北京大学出版社, 2001
2. 尤晋元,史美林等. Windows 操作系统原理. 北京: 机械工业出版社, 2001
3. Tanenbaum Andrew S. Modern Operating Systems(2nd Edition). Prentice Hall, 2001 (陈向群)

qianrushishi jisuanji

嵌入式计算机 (embedded computer) 作为一个信息处理部件,嵌入到应用系统中的计算机。

嵌入式计算机可以是微型计算机、工作站、小型计算机,甚至巨型计算机。但用得最多的还是单片计算机和单板计算机。一个应用系统可嵌入一台或多台计算机。由于嵌入式计算机埋藏在应用系统之中,一般不单独运行,甚至不暴露在现场操作人员的面前,所以在技术上具有如下特征:

(1) 在功能和形体结构上都嵌入于应用系统之中,其体积、重量、外形尺寸等物理参数要适应所嵌入的应用系统所提供的条件。其性能参数要符合所嵌入的应用系统的需要。

(2) 高可靠性 嵌入式计算机一般不进行日常检修,有的还运行于恶劣环境,所以对可靠性要求很高,例如宇宙飞船和卫星上的嵌入式计算机要能经得起发射时的加速度、振动以及太空的恶劣工作环境。在元器件测试、机械结构设计、热设计、软件和硬件可靠性设计等方面要求十分严格。为提高可靠性,广泛使用了冗余技术,以保证部分硬、软件出故障后,系统整体仍能正确运行。

(3) 自我检测功能 在应用系统工作时,往往不允许对嵌入式计算机进行维护。因此,在设计嵌入式计算机时,除了尽可能保证高可靠性外,还应考虑自我检测功能。要求嵌入式计算机能自动、迅速、准确地对故障进行检测、定位。有容错功能的还应根据故障情况对计算机系统进行重构。必要时,亦应便于对嵌入式计算机进行整体更换,以尽可能减少因嵌入式计算机故障而影响整个应用系统的正常

工作。

(4) 实时性 嵌入式计算机运行时有很强的实时性约束。嵌入式计算机大多从传感器接收输入信号,经处理后,其输出信号用于驱动各种执行机构,或显示出来供实时决策参考。

(5) 通用机专用 在一个特定的应用系统中,嵌入式计算机往往承担固定不变的专用任务。可以按应用系统的特定要求来设计专用计算机,但这是很不经济的。较好的办法是对通用计算机作适当的剪裁来满足不同应用系统的特定要求。

(6) 软件固化 嵌入式计算机软件在某一特定的应用系统中往往固定不变,虽然可以修改和扩充,但一旦投入使用,将长时间不变,所以嵌入式计算机的软件常被固化。

发 展 简 史

按照通用化、标准化、系列化的程度,嵌入式计算机的发展可分为3个阶段。

第一阶段(20世纪70年代中至80年代初)为初始阶段,基本上属于军用标准阶段。在这个阶段,标准化工作主要着眼于指令系统结构等硬件措施上,虽然美国各军兵种所使用的嵌入式计算机开始时曾有标准的军用指令系统结构,但各军兵种并未统一,形成了陆军的 NEBULA 结构;海军的 UYK-43,UYK-44,EMS P 以及 AYK-14 结构;空军的 MIL-STD-1750 A 结构。使用的编程语言也不统一,如陆军使用 NEBULA,空军使用 JOVIAL 和 PASCAL,海军使用 CMS-2 等。由于应用程序互不通用,影响了互相的交流。这个阶段的嵌入式实时操作系统一般都较简单,是为某种特定应用而设计的专用监控程序。直至80年代初才有了对应于某种微处理器芯片而设计的通用实时管理程序,如 Hunter & Ready Inc. 的 VRTX 和 Software Component Group Inc. 的 PSOS 等。它们主要用于以 Intel 8086 和 MC 68000 系列为中央处理器的嵌入式计算机系统。如美国空军的 MIL-STD-1750 A 嵌入式计算机采用了 VRTX 实时管理程序。

第二阶段为80年代中后期。在这个阶段,嵌入式计算机的通用化、标准化、系列化有了重大进展。从1984年起,美国国防部规定军内软件统一采用 Ada 语言编程,另外开发了 STARS 技术以补充 Ada 语言。所以在软件方面推行 Ada 语言和 STARS 技术,在硬件方面推行各种总线标准和接口标准是这个阶段的主要特征。这时嵌入式实时操作系统已趋完善,如 Real-time Performance 公司的 PRCore, Digit-

al Research 公司的 Flex OS 等。军用嵌入式计算机除了使用军用标准计算机外,也使用民用计算机,以保持和现代新技术同步。另外,美国在推行 Ada 语言的基础上,提倡先写应用软件,然后根据应用的需要再来配置计算机系统的硬件和软件,从而保证嵌入式计算机硬、软件技术的先进性。

第三阶段为80年代末以后,嵌入式计算机的应用更加普及,市场迅速扩大。这个时期嵌入式计算机硬件方面的显著特征是采用高性能的32位 CISC (参见复杂指令集计算机)或 RISC (参见精简指令集计算机) 处理器芯片,如 80386, 80486, i960, SPARC, R 2000, R 3000, AMD 2900 等。而软件方面的特点是面向开放系统、多处理和集成开发工具,如 Software Component Group Inc. 推出了面向32位 RISC, CISC 处理器的嵌入式实时操作系统 PSOS + m, 它保留了 PSOS 的特性,并具有多处理功能和丰富的开发工具, Intel 公司也针对其32位处理器,推出了嵌入式实时操作系统内核 iRMK III 以支持 80386, 80486 和 i960。

分 类 与 应 用

嵌入式计算机的分类方法很多,按应用领域可分为军用计算机、工业用计算机和民用计算机。通常,军用计算机和工业用计算机的运行环境有特殊的要求(如对温度、湿度、振动、冲击、灰尘、腐蚀、辐射等),为了适应所嵌入系统的各种不同的恶劣环境条件,嵌入式计算机又可分为加固计算机、半加固计算机和未加固计算机,按照嵌入应用系统的计算机本身类型可以分为微型计算机、工作站、小型计算机、大型计算机等。嵌入式微型计算机又可分为单片计算机、单板计算机和多板计算机等。

计算机的嵌入式应用已成为计算机应用的主要形式之一,特别是单片计算机已广泛用于家用电器(如电冰箱、自动洗衣机、照相机、空调机等),仪器仪表,医疗设备,数控机床,工业机器人,战略战术武器系统以及航天、测控和导航系统等。(沈祖恩)

Qiaomusiji cengci

乔姆斯基层次 (Chomsky hierarchy) 由 N. Chomsky 提出的形式语言的分层(参见形式语言理论)。它研究四类文法,即短语结构文法、上下文有关文法、上下文无关文法和正则文法之间存在的层次关系,以及由该四类文法产生的语言族所形成的层次。

短语结构文法 一个四元组 $G = (\Sigma, V, S, P)$,

其中 V 是非终结符的有穷字母表, Σ 是终结符的有穷字母表, P 是生成式的有限非空集, P 中的生成式均为 $\alpha \rightarrow \beta$ 的形式, $\alpha \in (\Sigma \cup V)^+$, $\beta \in (\Sigma \cup V)^*$, $S (\in V)$ 称为开始符号。短语结构文法是一种非受限文法, 也称为 0 型文法和半丘系统。对短语结构文法中的生成式作某些限制, 即得到上下文有关文法、上下文无关文法和正则文法。

上下文有关文法 一种特殊的短语结构文法, 也称为 1 型文法。它限制 P 中的生成式均为 $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$ 的形式, 其中 $A \in V$, $\alpha_1, \alpha_2, \beta \in (\Sigma \cup V)^*$, $\beta \neq \varepsilon$ 。对这类文法, 在派生中只有“上下文” α_1, α_2 分别出现在 A 的左、右邻近时, 才允许用 β 替换 A , 故云“上下文有关”。

上下文无关文法 一类特殊的上下文有关文法, 也称为 2 型文法。它限制 P 中的生成式均为 $A \rightarrow \beta$ 的形式, 其中 $A \in V$, $\beta \in (\Sigma \cup V)^*$, $\beta \neq \varepsilon$ 。

正则文法 一类特殊的上下文无关文法, 也称为 3 型文法。这类文法要求 P 中的生成式均为 $A \rightarrow aB$ 或 $A \rightarrow a$ 的形式, 其中 $A, B \in V$, $a \in \Sigma$ 。在有些文献中也称左线性文法或右线性文法为正则文法。它们都是和正则文法等价的。称文法是左(右)线性文法, 如果 P 中的生成式都是 $A \rightarrow Bw$ ($A \rightarrow wB$) 或 $A \rightarrow w$ 的形式, 其中 $A, B \in V$, $w \in \Sigma^*$ 。

由 i 型文法 $G = (\Sigma, V, S, P)$ 产生的语言称为 i 型语言, $i = 0, 1, 2, 3$, 记作 $L(G)$, 即

$$L(G) = \{x \in \Sigma^* \mid S \xRightarrow{*} x\}.$$

从四类文法的定义可见, 它们之间存在明显的层次关系, 从而, 四个语言类也存在层次关系。若令 \mathcal{L}_i 表示 i 型语言类, $i = 0, 1, 2, 3$, 则 $\mathcal{L}_3 \subseteq \mathcal{L}_2 \subseteq \mathcal{L}_1 \subseteq \mathcal{L}_0$ 。由于 $L = \{a^n b^n \mid n \geq 1\} \in \mathcal{L}_2$, 但不是 3 型语言, 故 \mathcal{L}_3 是 \mathcal{L}_2 的真子集; 又由于 $L = \{a^i \mid i \geq 1\} \in \mathcal{L}_1$, 但不是 2 型语言, 所以 \mathcal{L}_2 是 \mathcal{L}_1 的真子集。类似地, 在 \mathcal{L}_0 中也存在不属于 \mathcal{L}_1 的语言, 故 \mathcal{L}_1 也是 \mathcal{L}_0 的真子集。这就是所谓的乔姆斯基层次。在有些文献中也称为乔姆斯基分层或乔姆斯基谱系。

关于语言的描述, 除了从生成的角度, 即文法的角度之外, 还可从识别的角度, 即自动机的角度来描述。相应于四类文法有四类自动机, 即图灵机、线性有界自动机、下推自动机和有限自动机。并证明了: 图灵机识别的语言类恰为 \mathcal{L}_0 ; 线性有界自动机识别的语言类恰为 \mathcal{L}_1 ; 下推自动机识别的语言类恰为 \mathcal{L}_2 ; 有限自动机识别的语言类恰为 \mathcal{L}_3 。有限状态

自动机和正则文法的等价性是由 N. Chomsky 和 G. A. Miller 给出的, N. Chomsky 和 J. Evey 证明了下推自动机和上下文无关文法的等价性。S. Y. Kuroda 证明了线性有界自动机与上下文有关文法的等价性。N. Chomsky 证明了图灵机与短语结构文法的等价性。

参考文献

Hopcroft J E, Ullman J D. Introduction to Automata Theory, Languages, and Computation. Addison Wesley Publishing Company, 1979 (苏锦祥)

Qiaomusiji fanshi

乔姆斯基范式 (Chomsky normal form)

一种由 N. Chomsky 提出的上下文无关文法(2 型文法)的规范化形式。它对上下文无关文法生成式的形式加以某种限制。乔姆斯基证明了任意的上下文无关文法都存在等价的乔姆斯基范式。

若上下文无关文法 $G = (\Sigma, V, S, P)$ 中的生成式均为 $A \rightarrow BC$ 或 $A \rightarrow a$ 的形式, 则称 G 为具有乔姆斯基范式的文法, 其中 $A, B, C \in V$, $a \in \Sigma$, 缩写为 CNF。N. Chomsky 在 1959 年证明了任何不含 ε 的上下文无关语言都可由乔姆斯基范式产生, 这就是著名的乔姆斯基范式定理。实际上, 乔姆斯基证明了更强的结果, 即证明了: 每个不含 ε 的上下文无关语言, 均可由一个如下所述的上下文无关文法 $G = (\Sigma, V, S, P)$ 产生, 其中所有的生成式均为 $A \rightarrow BC$ 或 $A \rightarrow a$ 的形式, $A, B, C \in V$, $a \in \Sigma$, $B \neq C$ 且若 $A \rightarrow \alpha_1 B \alpha_2$, $A \rightarrow \beta_1 B \beta_2$ 是 P 中的生成式, 则 $\alpha_1 = \beta_1 = \varepsilon$ 或 $\alpha_2 = \beta_2 = \varepsilon$, $\alpha_1, \alpha_2, \beta_1, \beta_2 \in (\Sigma \cup V)^*$ 。

根据乔姆斯基范式定理还可构造出一个等价的乔姆斯基范式, 这种范式不仅使相应的上下文无关文法的生成式异常简单而且规范。这对研究相应的上下文无关语言的结构很有意义。例如, 由乔姆斯基范式所产生的任意串对应的推导树都是一棵二叉树。

参考文献

Hopcroft J E, Ullman J D. Introduction to Automata Theory, Languages, and Computation. Addison Wesley Publishing Company, 1979 (苏锦祥)

Qiebixuefu bijin

切比雪夫逼近 (Chebyshev approximation)

又称最佳一致逼近(参见数值逼近)。

令度量 $p = \infty$, 且 $H = H_n$ 为次数不大于 n 的多项式函数集合。设 $f \in C[a, b]$, 若 $P \in H_n$ 满足

$$E_n(f) \equiv \inf_{Q \in H_n} \|f - Q\|_{\infty} = \|f - P\|_{\infty}$$

则称 P 为 f 的次数不大于 n 的切比雪夫逼近多项式函数, 称 $E_n(f)$ 为 H_n 对给定函数 f 的最小偏差。且有

$$E_0(f) \geq E_1(f) \geq \cdots, \quad \lim_{n \rightarrow \infty} E_n(f) = 0$$

事实上, 这样的多项式 P 是存在且惟一的。

切比雪夫定理: H_n 中的多项式函数 P 成为 $C[a, b]$ 中给定函数 f 的切比雪夫逼近多项式函数的充要条件是在 $[a, b]$ 上存在一组分点(称为偏差点组或交错点组)

$$a \leq x_1 < x_2 < \cdots < x_m \leq b \quad (m \geq n+2)$$

使得

$$f(x_i) - P(x_i) = (-1)^i \lambda$$

$$|\lambda| = \|f - P\|_{\infty} \quad (i = 1, 2, \cdots, m)$$

成立。

由定理知, 当 $n = 0$ 时, 零次最佳一致逼近多项式为

$$P(x) = \frac{m+M}{2}$$

其中 $m = \min_{a \leq x \leq b} f(x)$, $M = \max_{a \leq x \leq b} f(x)$

当 $n = 1$ 时, 若 f 在 $[a, b]$ 上的二阶导数 f'' 连续且不变号, 则一次最佳一致逼近多项式为

$$P(x) = a_0 + a_1 x$$

其中

$$a_1 = \frac{f(b) - f(a)}{b - a},$$

$$a_0 = \frac{f(a) + f(x_1)}{2} - a_1 \frac{a + x_1}{2}$$

x_1 是方程 $f'(x) = a_1$ 的根。

当 $n \geq 2$ 时, 可由定理得到的列梅兹算法来逐次逼近求得。但实际计算很困难, 一般很少使用。往往借助切比雪夫多项式 $T_n(x) = \cos(n \arccos x)$ 的性质来求近似的最佳一致逼近多项式。

使用三角多项式来逼近 $C_{2\pi}$ 中的函数时, 可得到完全平行的结果。

参考文献

Davis P J. Interpolation Approximation. Blaisdell, Waltham, Mass, 1963

(沈毅)

qingxipan

清洗盘 (cleaning disk) 清洁软磁盘驱动器磁

头的专用盘片。

软磁盘驱动器使用时, 其磁头和软磁盘片相互磨擦, 磨擦产生的磁粉和外界渗入的灰尘堆积并粘附在磁头表面, 造成读写错误, 若不及时清除, 可能导致划伤盘片。因此必须定期清洗磁头, 以保证驱动器和盘片的正常使用。驱动器结构复杂, 人工清洗磁头需要技术训练, 因而通常采用清洗盘清洗。

清洗盘的外形尺寸和记录用的软磁盘相同。有干式和湿式两种。干式清洗盘用精细砂纸制成, 使用时插入驱动器, 由主轴带动清洗盘旋转, 通过磨擦去掉磁头表面的污物。湿式清洗盘由特种薄纸制成, 使用时在盘上均匀敷滴专用清洁溶剂, 插入驱动器后, 同样由主轴带动清洗盘旋转, 靠与磁头的磨擦以及溶剂的化学作用, 去掉磁头上的污物。两种清洗盘都有一定的寿命(使用次数), 每次使用时要改变使用的位置(不同的磁道位置), 以达到良好的清洗效果。另外, 清洗时间不能太长, 湿式清洗盘使用时加入的清洁溶剂不能太多, 以免损伤磁头。

当驱动器磁头太脏时, 用清洗盘很难清洗干净, 应由经过专门训练的技术人员进行人工清洗。

(王国元)

qumian

曲面 (surface) 曲线上所有点运动产生的轨迹。曲面一般分为解析曲面和不能用初等解析式表示的自由型曲面。在计算机图形学中, 曲面是表示和构造复杂形体外形的重要数学工具。

解析曲面的显式或隐式表示历史悠久, 而适合计算机处理的参数曲面则在近几十年内得到迅速发展。1963年 Ferguson 用4个角点的位置矢量及其2个方向切矢量定义三次曲面。1964年 Coons 用封闭的4条边界曲线定义一块曲面。同年, Schoenberg 提出了参数样条曲面的形式。1971年 Bézier 提出了一种用控制多边形定义曲面的方法。1972年, de Boor 给出了B样条的标准计算方法。1974年, Gordon 和 Riesenfeld 将B样条理论用于形状描述, 提出B样条曲面。20世纪80年代后期, Piegl 和 Tiller 将有理B样条发展成非均匀有理B样条(NURBS)。当前, 它已成为描述曲面的广为流行的方法, 用NURBS可统一表示初等解析曲面、有理与非有理Bézier以及非有理B样条曲面。

对于曲面, 主要研究它的数学表示、连续性及其在实际中的应用, 重点研究用计算机构造具有一定连续性的实用曲面的理论与算法。

曲面的连续性指的是曲面片与曲面片之间的连接光滑程度。如果两曲面具有公共连接线,称它们是位置连续的或是 C^0 的,即具有零阶参数连续性。一般的曲面参数连续性即 C^n 连续性定义为:当且仅当两曲面 $p(s, t)$ 与 $q(u, v)$ 沿它们的公共连接线 $p(\gamma) = q(\gamma)$ 处处具有直到 n 阶的连续偏导矢,则称它们沿该连接线具有 n 阶参数连续性(即 C^n 连续性)或是 C^n 的。即

$$\frac{\partial^{i+j}}{\partial s^i \partial t^j} p(\gamma) = \frac{\partial^{i+j}}{\partial u^i \partial v^j} q(\gamma), \quad i+j=1, 2, \dots, n$$

两参数曲面的零阶几何连续性即 G^0 连续性是与 C^0 连续性一致的。

两参数曲面的 G^1 连续性又称为切平面连续性,其定义为:两曲面沿它们的公共连接线具有 G^1 连续性或是 G^1 的,当且仅当它们沿该公共连接线处处具有公共的切平面或公共的曲面法线。

两参数曲面的 G^2 连续性又称为曲率连续性,其定义为:两曲面沿它们的公共连接线具有 G^2 连续性或是 G^2 的,当且仅当它们沿该公共连接线处处具有公共的切平面外,又具有公共的主曲率,并在两个主曲率不相等时具有公共的主方向,或一致的杜潘标线。

一般的曲面几何连续性即 G^n 连续性定义为:两曲面 $p(s, t)$ 与 $q(u, v)$ 沿它们的正则公共连接线具有 G^n 连续性或是 G^n 的,当且仅当其中之一,譬如 q , 可被重新参数化为 $\tilde{q}(\tilde{u}, \tilde{v})$, 以致于它们沿该公共连接线是 C^n 的。即

$$\frac{\partial^{i+j}}{\partial s^i \partial t^j} p(\gamma) = \frac{\partial^{i+j}}{\partial \tilde{u}^i \partial \tilde{v}^j} \tilde{q}(\gamma), \quad i+j=1, 2, \dots, n$$

在曲面造型中(参见几何造型),一般只用到 C^1 , C^2 和 G^1 , G^2 连续。

参考文献

1. Farin G. Curves and Surfaces for Computer Aided Geometric Design. Academic Press Inc., 1993
2. Mortenson M E. Geometric Modeling. John Wiley & Sons, 1985
3. 孙家广, 杨长贵. 计算机图形学(新版). 北京: 清华大学出版社, 1995 (孙家广 冯结青)

quxian

曲线(curve) 一个点在空间运动的轨迹。曲线一般分为解析曲线和不能用初等解析式表示的自由型曲线。在计算机图形学中,曲线是表示和构造复杂形体外形的重要数学工具。

解析曲线的显式或隐式表示历史悠久,而适合

计算机处理的参数曲线则在近几十年内得到迅速发展。1963年 Ferguson 用三次参数法构造曲线。同年, Schoenberg 提出了参数样条曲线的形式。1971年 Bezier 提出了一种用控制多边形定义曲线的方法。1972年, de Boor 给出了 B 样条的标准计算方法。1974年, Gordon 和 Riesenfeld 将 B 样条理论用于形状描述,提出 B 样条曲线。20 世纪 80 年代后期, Piegl 和 Tiller 将有理 B 样条发展成非均匀有理 B 样条(NURBS)。当前,它已成为描述曲线的广为流行的方法,用 NURBS 可统一表示初等解析曲线、有理与非有理 Bezier 以及非有理 B 样条曲线。

对于曲线,主要研究它的数学表示、连续性及其在实际中的应用,重点研究用计算机构造具有一定连续性的实用曲线的理论与算法。

曲线的连续性指的是曲线段与曲线段之间的连接光滑程度。由调和函数构造的参数曲线,其自身(即在参数 t 的取值区间内)的连续性是由调和函数决定的。而对于几条参数曲线段首尾相接构成的参数曲线,如何保证连接处具有合乎要求的连续性是关键问题。图 1 中给出 4 条参数曲线段 $Q_1(t)$ 和 $Q_2(t)$, $Q_3(t)$ 和 $Q_4(t)$, $t \in [0, 1]$:

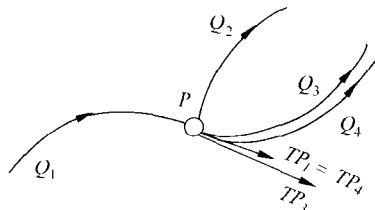


图 1 曲线段之间的连续性

(1) 若 $Q_1(1)$ 和 $Q_2(0)$ 的端点重合于 P , 则 $Q_1(t)$ 和 $Q_2(t)$ 在 P 点处有 C^0 和 G^0 连续。

(2) 若 $Q_1(1)$ 和 $Q_3(0)$ 在 P 点处重合, 且其在 P 点处的切矢量方向相同, 大小不等, 则 $Q_1(t)$ 和 $Q_3(t)$ 在 P 点处有 G^1 连续。

(3) 若 $Q_1(1)$ 和 $Q_4(0)$ 在 P 点处重合, 且其在 P 点处的切矢量方向相同, 大小相等, 则 $Q_1(t)$ 和 $Q_4(t)$ 在 P 点处有 C^1 连续。

(4) 若 $Q_1(1)$ 和 $Q_4(0)$ 在 P 点处已有 C^0 , C^1 连续且其 $Q_1'(1)$ 和 $Q_4'(0)$ 的大小和方向均相同, 则 $Q_1(t)$ 和 $Q_4(t)$ 在 P 点处有 C^2 连续。推而广之, 若 $Q_1'(1)$ 和 $Q_4'(0)$ 在 P 点处的大小和方向均相同, 则说 $Q_1(t)$ 和 $Q_4(t)$ 在 P 点处有 C^n 连续。

(5) 若 $Q_1(t)$ 和 $Q_3(t)$ 在 P 点处已具有 G^0 , G^1

连续,且其 $Q_1'(1)$ 和 $Q_3'(0)$ 的方向相同,但大小不相等,则说 $Q_1(t)$ 和 $Q_3(t)$ 在 P 点处有 G^2 连续。

在曲线造型中(参见几何造型),一般只用到 C^1, C^2 和 G^1, G^2 连续。在实际应用中,应适当选择曲线段的连续性,使造型物体既能保证其光滑性,也能保证其美观性。

参考文献

1. Farin G. Curves and Surfaces for Computer Aided Geometric Design. Academic Press Inc., 1993
2. Mortenson M E. Geometric Modeling. John Wiley & Sons, 1985
3. 孙家广,杨长贵. 计算机图形学(新版). 北京:清华大学出版社,1995 (孙家广 冯结青)

quan pindai shuzi yinpin de bianma

全频带数字音频的编码(coding of full band digital audio)

对全频带数字声音信号进行数据压缩和编码处理的方法与技术。声音由许多不同频率的谐波所组成,谐波的频率范围称为声音的带宽。人耳可听见的各种声音,如音乐声、风雨声、汽车声等,其带宽为 20 ~ 20 kHz,它们称为全频带音频信号。全频带音频信号数字化之后的数据量很大。以 CD 盘片上所存储的立体声高保真数字音乐为例,1 小时的数据量大约是 635 MB。为了降低存储成本和提高通信效率(降低传输带宽),对全频带数

字音频进行数据压缩是十分必要的。

全频带数字音频的数据压缩也是完全可能的,其依据是声音信号中包含有大量的冗余信息,再加上还可以利用人的听觉感知特性,因而产生了许多压缩算法。一个好的数据压缩算法应做到压缩倍数高,声音失真小,算法简单,编码器和解码器的成本低。

全频带数字音频的第一代压缩编码技术采用的是 PCM(脉冲编码调制)编码,它主要是依据声音波形本身的信息相关性进行数据压缩,代表性的应用是 CD 唱片。

第二代全频带音频的压缩编码不但充分利用声音信息本身的相关性,而且还充分利用人耳的听觉特性,即使用“心理声学模型”来达到大幅度压缩数据的目的,这种压缩编码方法称为感知音频编码。

图 1 是第二代全频带音频编码的基本原理。编码过程一般分为 3 个阶段,第一阶段通过时间频率变换和心理声学分析,揭示原始声音中与人耳感知无关的信息,然后在第二阶段通过量化和编码予以抑制,第三阶段再使用熵编码消除声音信息中的统计冗余。

表 1 是几种典型的第二代全频带音频的压缩编码标准。其中 MPEG-1 音频压缩编码是国际上第 1 个高保真音频数据压缩的国际标准,它分为 3 个层

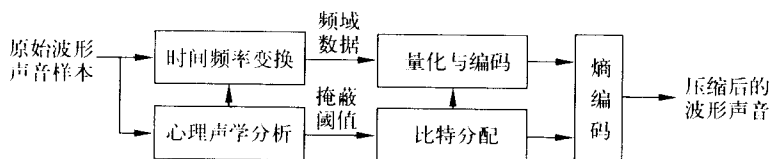


图 1 感知音频编码的原理框图

表 1 第 2 代全频带音频压缩编码标准

名称	每个声道压缩后的码率/kb/s	声道数目	主要应用
MPEG-1 层 1	192 (压缩 4 倍)	2	数字盒式录音带
MPEG-1 层 2	128 ~ 96 (压缩 6 ~ 8 倍)	2	DAB, VCD, DVD
MPEG-1 层 3	77 ~ 64 (压缩 10 ~ 12 倍)	2	Internet, MP3 音乐
MPEG-2 audio	与 MPEG-1 层 1, 2, 3 相同	5.1, 7.1	同 MPEG-1
MPEG-2 AAC	64 ~ 48 (压缩 12 ~ 16 倍)	48 个主声道, 16 个低频通道	DVD, DTV, 家庭影院
Dolby AC-3	64	5.1, 7.1	DVD, DTV, 家庭影院

次:层1的编码较简单,主要用于数字盒式录音磁带;层2的算法复杂度中等,其应用包括数字音频广播(DAB)和VCD等;层3的编码较复杂,主要用于因特网上高质量声音的传输。最近几年流行起来的MP3音乐,就是采用MPEG-1层3编码的数字音乐,它以10倍左右的压缩比降低高保真数字音频的数据量,一张普通CD光盘上可以存储大约100首MP3歌曲。

MPEG-2有2种音频压缩编码。第1种(称为MPEG-2 audio)采用与MPEG-1相同的方案,层1、层2和层3的结构也相同,但它能支持5.1声道和7.1声道的环绕立体声。第2种(称为MPEG-2 AAC)采用更先进的方案,效果也更好。

杜比数字AC-3(Dolby Digital AC-3)是美国杜比公司开发的多声道全频带音频编码系统,目前在数字电视、DVD和家庭影院中已得到广泛使用。

参考文献

Pohlman Ken C. Principles of Digital Audio, Fourth Edition. McGraw-Hill Companies, Inc., 2000

(张福炎)

quanqiu dingwei xitong

全球定位系统(global positioning system,

GPS) 利用人造地球卫星的信号准确测定待定点位置的全球导航系统。美国国防部为适应军事需要,从1973年开始研制全球定位系统(GPS),耗资300亿美元,于1993年完成全球覆盖率高达98%的24颗卫星的发射,建成这一号称第三大航天计划的宏伟工程,完全实现了人类在地球上的导航和定位。它的出现给全球导航的精密定位引入了崭新的技术,也带来了巨大的经济效益和社会效益。

全球定位系统(GPS)由三大部分组成:①空间部分,美国的GPS由24颗卫星组成,其中工作卫星21颗,备用卫星3颗,它们均匀分布在6个相互夹角为 60° 的轨道平面内,即每个轨道上有4颗卫星,保证了地球上任何地点在任何时间都可以至少同时观测到4颗卫星,形成全球性、全天候、三维定速定位的定位系统。②地面控制部分,美国的GPS由一个主控站(位于美国科罗拉多州)、3个注入站(分别位于太平洋的卡瓦加兰岛、印度洋的狄哥·伽西亚岛和大西洋的阿松森岛)、5个监测站(除位于上述4地外,再加上夏威夷群岛)以及通信辅助系统组成。进行GPS定位的一个先决条件是用户必须知道观测瞬间GPS卫星在空间的位置。由于不同用

户是在不同时间对不同的卫星进行观测,所以实际上是要知道所有的GPS卫星在任一时刻的位置,这是在全球定位系统的地面控制部分获得解决的。③用户装置部分,由GPS信号接收机和卫星天线组成。GPS信号接收机的任务是捕获GPS卫星发射的信号,并进行处理。根据信号到达接收机的时间,确定接收机到卫星的距离。如果计算出4颗或者更多卫星到接收机的距离,再参照卫星的位置,就可以确定接收机在三维空间中的位置,从而达到定位的目的。

美国为了防止用户未经许可将GPS用于军事目的,卫星发射采用两种编码:P码(也称精码)和C/A码(也称粗码)。P码的码率为10.23 MHz,周期为360 d,码元长度为29.3 m,定位精度为20 m,服务对象主要是美国军事部门;C/A码的码率为1.023 MHz,周期约为1 ms,码元长度为293 m,定位精度为100 m,一般服务于民用目的。

GPS的主要用途是:①陆地应用,主要包括车辆导航、地球观测、地球物理勘探、地壳运动监测、地质灾害预报、大地控制网建立、工程测量、出租车调度与管理等;②海洋应用,包括船舶调度与导航、海面变化监测、海洋石油钻井定位、海洋救援、海洋资源探测等;③航空航天应用,包括飞机导航、航空遥感姿态控制、导弹制导、低轨卫星定轨、航空救援等。

我国自20世纪80年代起,已在一些部门和地区开展GPS定位应用。90年代初开始着手布设包括700个点的全国GPS网,与此同时由4个在全国均匀分布的GPS跟踪站所组成的试验性跟踪网也在建设之中。先后研制了GPS定位的硬、软件设备,例如1996年在我国发射的返回式遥感卫星上,首次搭载自行研制的GPS自控定位接收机进行试验,获得成功。此外,还开展了减弱定位误差影响及动态GPS定位和无地面控制点成图技术等试验研究工作。

参考文献

1. 陈述彭主编. 地球系统科学. 北京: 中国科学技术出版社, 1998

2. 周忠谟, 易杰军. GPS全球定位系统原理及其应用. 北京: 测绘出版社, 1997 (黄杏元)

quanqiu yidong tongxin xitong

全球移动通信系统(Global System for Mobile communication, GSM) 一个全球范围的

数字移动通信标准体系。GSM 原意是指“移动通信特别小组”(Group Special Mobile)。它是 1982 年欧洲邮政电信管理委员会 (CEPT) 为开发第二代数字蜂窝移动系统而成立的机构。1982 年至 1983 年该机构着重讨论了 GSM 的基本原理;1984 年成立了三个工作组分别研究业务,制定无线传输规定和定义网络结构及开放接口的信令规程;1986 年 GSM 的永久研究机构成立;1987 年确定 GSM 的主要无

线传输技术;1988 年欧洲电信标准协会 (ETSI) 成立,GSM 转入这个新机构下;1991 年 GSM 第 1 阶段的技术规范全部完成,同年底,世界上第一个 GSM 网络开始运营;1995 年 GSM 第 2 阶段的技术规范全部完成。

GSM 网络结构如图 1 所示,主要由网络子系统 (NSS)、基站子系统 (BSS) 和移动台 (MS) 三部分组成。

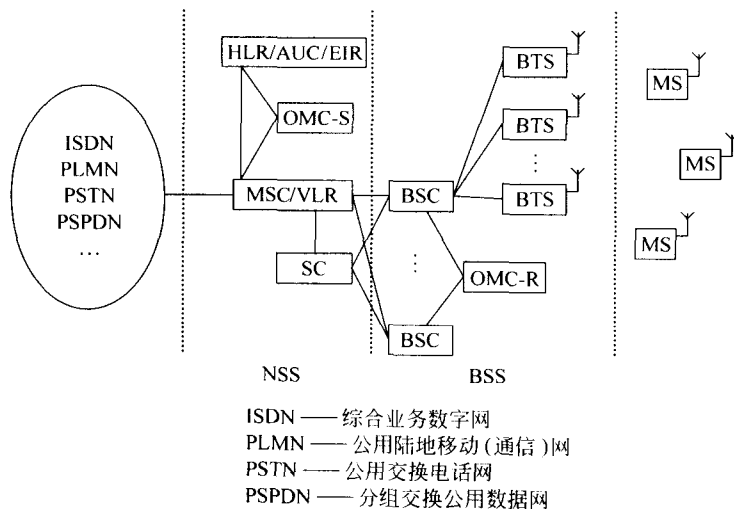


图 1 全球移动通信系统 (GSM) 网络构成

移动台 (MS) 是用户使用的终端设备。它包括移动电话以及用于数据-传真等附加业务的终端适配器和终端设备。其主要功能除了通过无线接入进入通信网络,完成各种控制和处理以提供主叫或被叫通信以外,还提供与使用者之间的人机接口或与其他终端设备连接的适配装置等。移动台通过用户身份模块 (SIM) 卡向通信网络提供了用户注册和管理所需的信息。

基站子系统 (BSS) 包含了 GSM 无线通信部分的所有地面基础设施,是移动台和交换机之间的桥梁。BSS 分为三个部分:基站控制器 (BSC)、基站收发信机 (BTS) 以及操作维护中心 (OMC - R)。拥有 BSC 是全球移动通信系统和模拟蜂窝系统的一个重要区别。一个 BSC 可控制多个 BTS。BSC 的主要功能是管理无线资源 (无线信道的分配、释放、切换、功率控制、跳频等),它还能将多个 BTS 的话务进行集中。BTS 包括无线收发信机和天线。此外还有与无线接口有关的信号处理电路,负责在一个小区内进行无线信号的转发,并将呼叫送至 BSC。OMC - R

是对 BSS 进行操作和维护的功能实体。

网络子系统 (NSS) 由移动交换机 (MSC)、归属位置寄存器 (HLR)、访问位置寄存器 (VLR)、鉴权中心 (AUC)、设备识别寄存器 (EIR)、操作维护中心 (OMC - S) 和短消息业务中心 (SC) 组成。移动交换机 (MSC) 是对位于它覆盖区域中的移动台进行控制和交换话务的功能实体,也是 GSM 网络与其他通信网之间的接口实体,负责整个 MSC 区内的呼叫控制、移动性管理和无线资源管理。访问位置寄存器 (VLR) 是存储进入覆盖区用户与呼叫处理有关信息的动态数据库。MSC 为了处理位于本覆盖区中移动台的来话和去话呼叫需到 VLR 检索信息,通常 MSC 与 VLR 合设于同一物理实体中。归属位置寄存器 (HLR) 是用于移动用户管理的数据库。每个移动用户都应在其归属的位置寄存器注册登记。HLR 主要存储两类信息:一类是有关用户的业务信息,另一类是用户的位置信息。鉴权中心 (AUC) 存储用以保护移动用户通信不受侵犯的必要信息。设备识别寄存器 (EIR) 是存储有关移动台设备参数的

数据库,主要完成对移动设备的识别、监视和闭锁等功能,用来确保非法移动台不能使用。操作维护中心(OMC-S)是执行NSS操作维护的功能实体。短消息业务中心(SC)是用于向用户提供短消息业务的实体。HLR、AUC和EIR通常合设于同一物理实体中。

与其他制式的蜂窝系统相比,GSM在网络接口的规范化方面具有明显的优势。GSM系统定义了Um接口、Sm接口、A接口、Abis接口、B、C、D、E、F、G等多个接口,其中B~G都是由移动应用部分(MAP)支持的,所以又统称为MAP接口。这些接口的标准化为实现不同生产厂家产品的互连提供有利的保障。

按照传输信息的特性,GSM移动通信网向用户提供的业务可以划分为两类:语音(言语)业务和数据业务。按照ISDN的观点,可划分为:承载业务、用户业务、补充业务。语音(言语)业务包括电话、紧急呼叫、语音(言语)信箱等业务。数据业务中包括传输除语音(言语)之外的其他各种信号,包括文字、图像、传真、短消息和计算机的文件等。承载业务是指通信网络向用户提供的信息传送业务,包括OSI 1~3层功能。用户业务是指用户在通信网络上通过终端设备能够获得的业务,包括OSI 1~7层功能。补充业务是指通信网络在向用户提供基本业务的同时,为了使用户通信更加便利以及提高网络利用率而附加的业务。

参考文献

1. 孙孺石,丁怀元,穆万里,王泽权. GSM 数字移动通信工程. 北京:人民邮电出版社,1996
2. 王志勤,魏然,王小云. 全球移动通信系统(GSM)的技术及发展. 电信科学,1996,12(4)

(史美林 英春)

quanwen jiansuo

全文检索(full-text retrieval) 把文献中出现的每一个词(或字)都作为检索入口的基于全文标引的检索过程和技术。在全文检索系统中,文献中任何有检索意义的词或字串都可被检索出来。可以用布尔逻辑进行组合检索;可以用位置逻辑进行组合检索;可以使用停用词技术剔除文献中无检索意义的词,留下有实际意义的词作为检索入口而加以标引。若只能根据文献的外部特征找出全文,或只提供布尔逻辑运算,而没有提供位置逻辑运算,都不能称作全文检索。

全文检索与传统的人工标引检索相比较,具有以下优点:①自动建立数据库,速度快;②不存在词汇滞后问题;③不损失专指性;④不产生漏检;⑤可以直接提供原文献。全文检索主要分为基于关键词匹配的精确检索和根据内容的概念检索两类。在实现技术上,全文检索采用的算法主要有:①全文扫描;②倒排文件;③位图文件。

为了提高全文检索的结果质量,采用相关排序与相关反馈等技术。全文检索的扩展包括能利用文字来检索多媒体信息,结合超文本技术及通过交互式的浏览和导航来改善检索的效果。

全文检索的缺点是:①索引所占的物理空间大;②检索速度慢;③建立索引的时间与内存开销大。但随着计算机软、硬件的发展及数据压缩技术的成熟,全文检索技术已经完全实用,因特网上的各种资源数据库普遍采用先进的全文检索技术。

中文全文检索可分为按字全文检索与按词全文检索。按词全文检索具有检索速度快、查准率高、空间开销小等优点。同时,如果要利用较为高级的检索技术,如相关排序,则按词建库和检索具有较大的优越性。

1959年美国法律界在匹兹堡大学建立了第一个全文检索系统。广域信息服务系统是新一代全文检索系统的典型。

参考文献

1. Salton G, McGill M J. Introduction to Modern Information Retrieval. McGraw Hill, 1983
2. Salton G. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison Wesley, 1989
3. Frakes W B, Yates R B. Information Retrieval: Data Structures and Algorithms. Prentice Hall, 1992

(施水才)

quanxi zhaoxiang cunchuqi

全息照相存储器(holographic memory) 以激光器为光源,利用全息照相的方法,实现数字信息存储的装置。全息照相存储机理可简单描述为:待存储的数据经空间光调制器(SLM)形成一个二维信息页加载到信号光上,然后与参考光在记录介质中发生干涉,利用材料的光折变效应形成体全息光栅,从而完成信息的记录;读出时使用和原来相同的参考光寻址,可以读出相应的存储在晶体中的全息图,然后使用光信号探测器件(CCD)对参考光读出的

图像进行采集。根据体全息图的布拉格角度选择性或者布拉格波长选择性,改变参考光的入射角度或波长可以实现多重存储。体全息存储可以在一个单位体积内复用多幅图像,实现超高密度存储。

1963年美国科学家首次提出利用全息技术进行数据存储,在其后四十多年的时间内许多研究机构都做出了很大贡献,特别是在20世纪90年代,随着计算机科学和现代信息处理技术的不断发展,体全息存储技术有了很大的突破。1995年美国实施了光折变信息存储材料(PRISM)项目和全息数据存储系统(HDSS)项目,目前一些研究机构正在致力于全息存储的商品化。

体全息存储主要利用光折变效应进行存储。为了在晶体的同一体积内存储尽量多的信息,实现超高密度存储,就要进行复用。复用一般是指在晶体中的同一体积内存储多幅全息图的方法。体全息存储的另外两个重要参数是传输速率和误码率,信号输入输出设备(I/O)的性能是存储系统数据传输量和传输速率的决定性因素。SLM和CCD器件(包括CMOS器件)的发展为充分发挥体全息存储的优点提供了硬件基础,使之成为新一代存储技术的代表。

体全息技术具有一些其他技术所不能比拟的特性,其中包括存储密度高、数据并行传输、数据冗余度高、寻址速度快、具有关联寻址功能等。它不但可以用作数字体全息存储器,还可以用于一些光学处理系统(如星载存储器)和图像识别系统,它对于巡航导弹防卫、导航自动目标识别、卫星星图匹配定位、大型数据库的检索与管理等应用都十分重要。体全息存储技术跨越了信息存储和信息处理两个领域,随着各种相关技术(如光源、材料和关键光电器件)的发展,体全息存储必将在计算机和光学信息处理中发挥重要作用。

参考文献

1. Yeh P, Gu C. Landmark papers on Photorefractive Nonlinear Optics. New Jersey: World Scientific, 1995
2. Coufal H J, Psaltis D, Sincerbos G T. Holographic Data Storage. New York: Springer-Verlag, 2000
(何庆声 金国藩)

quanbiao huanwang

权标环网(token ring network) 采用权标传递方法进行介质访问控制的一种环状局域网。在这

种环状拓扑结构中,通信信道由一组用点到点的方式将传输介质连接为闭合环的插入器组成。每个插入器都与两条链路相连。站点通过网络适配器、经插入器连线与插入器相连。其物理结构如图1所示。插入器接收一条传输介质上的数据,并以同样的速率串行地将该数据传送到另一条传输介质上。所有传输介质都按同一方向传输,因此环形网中没有路径选择问题。

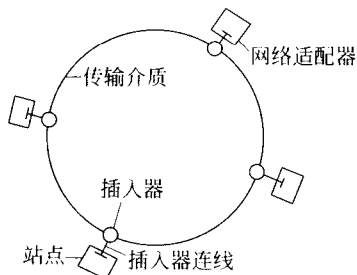


图1 权标环网的物理结构图

IEEE 802.5(参见局域网协议标准)是用于权标环网介质访问控制的标准,规定了数据帧格式和权标传递方法。在权标环网上所有站点都分配有逻辑地址,且每个站点都知道自己下游的站点。权标是一种特殊帧,在环网上只允许一个权标帧传递。环网上的每个站点都有机会获取权标而发送信息帧。当一个站点既想发送信息帧又获得权标,该信息帧就会在环网上传输。所有其他站点都监视该信息帧里的目的地址是否与自己站点地址相符。若不符,则让信息帧传输到下游站点;若相符,则该站点就将该信息帧拷贝下来,同时让该帧继续在环网上传输,直至返回到源发站点。源发站点收到该帧后,就释放所持权标,让权标继续在环网上传递,以便让其他需要发送信息帧的站点获取。权标帧和信息帧的格式如图2所示。

信息帧:

字节数	1	1	1	2/6	2/6	≥ 0	4	1	1
	SD	AC	FC	DA	SA	数据	FCS	ED	FS

SD=起始定界符;AC=访问控制;FC=帧控制;
DA=目的地址;SA=源地址;FCS=帧检验序列;
ED=结束定界符;FS=帧状态

权标帧:

字节数	1	1	1
	SD	AC	FS

SD=起始定界符;AC=访问控制;FS=帧状态

图2 权标环网 IEEE 802.5 的帧格式

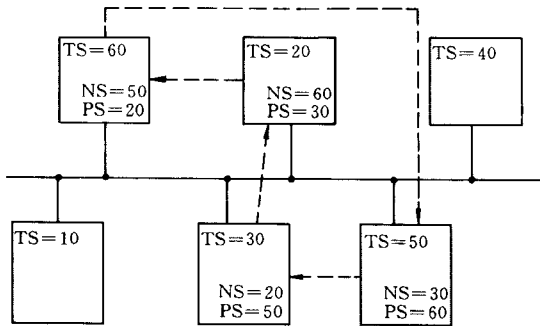
参考文献

Tanenbaum A S. Computer Networks, 3/e, Prentice Hall Inc., 1996 (徐伟氏)

quanbiao zongxian wang

权标总线网 (token bus network) 采用权标传递方法进行媒体访问控制的一种总线局域网。其传输速率为 5 Mb/s 或 10 Mb/s, 传输媒体采用 75 Ω 同轴电缆, 媒体上传输的信号为经过调制的数字信号。

在权标总线网上, 各站点通过网络适配器连成一线总线结构。如图 1 所示。各站按其网络适配器上所设的物理地址大小排成一个逻辑环。在逻辑环中, 每个站点必须知道其前一个和后一个站点的地址, 最大地址站点排在最小地址站点之后。具有目的地址和源地址的权标帧, 从一个站点按地址顺序传递到下一个站点。若某个站点的信息帧已准备好, 一旦它接收到权标, 就向媒体上发送信息帧。等到该站点中所有排队等候发送的信息帧均发出或权标持有时间超过一定限值后, 权标被传送给下一个站点。这就是权标总线网的媒体访问控制方法, 它比较复杂, 要花费较多的开销对逻辑环进行维护。



TS —— 本站点地址 NS —— 下一站点地址
PS —— 前一站点地址
图 1 权标总线网

权标总线网具有实时性和优先访问机制, 因而常用在工业控制环境中。

国际标准化组织 (ISO) 曾对权标总线网制定了 1 个国际标准, 其标准号为 ISO 8802—4。

(张公忠)

qun

群 (group) 一种具有一个二元运算的代数

结构。

若非空集合 G 及其上的二元运算 $*$ 满足结合律, 即对任意 $a, b, c \in G$ 有 $a * (b * c) = (a * b) * c$, 则称 G 为一个半群, $*$ 为半群 G 的乘法, $a * b (a, b \in G)$ 为 a 与 b 的积, 常简写为 ab 。

由归纳法可证明, 对任意 $a_1, \dots, a_n \in G (n \geq 1)$ 有惟一确定的积 $a_1 \cdots a_n$, 即只要 a_1, \dots, a_n 的次序不变, 无论在它们间怎样添加括号, 结果都一样。这就是所谓的广义结合律。

若 $e \in G$ 使得对任意 $a \in G$ 有 $ae = a = ea$, 则称 e 为半群 G 的一个幺元或单位元。具有幺元的半群称为幺半群。

全体正整数关于普通的加法和乘法分别构成半群和幺半群。全体实 $n (n \geq 1)$ 阶方阵关于矩阵乘法构成幺半群。

如果半群 G 满足以下公理:

(1) 存在左幺元, 即对每个 $a \in G$ 皆有 $ea = a$;

(2) 存在左逆元, 即对每个 $a \in G$, 皆有 $a^{-1} \in G$ 使 $a^{-1}a = e$ 。则称 G 为一个群, 记为 $\langle G, * \rangle$ 。

可以证明, 对群 G 中任意元素 a 有 $ae = a$ 和 $aa^{-1} = e$, 而且 e 和 a^{-1} 还都是惟一的。因此称 e 为 G 的幺元或单位元, a^{-1} 为 a 的逆元。此外, 对任意 $a, b \in G$ 显然有 $(ab)^{-1} = b^{-1}a^{-1}$ 。

根据广义结合律, 对任意 $a \in G$ 和正整数 n , 可令

$$a^0 = e, \quad a^n = \underbrace{aa \cdots a}_n, \quad a^{-n} = (a^{-1})^n$$

于是, 对任意整数 m, n 有 $(a^m)^n = a^{mn}$ 和 $a^m a^n = a^{m+n}$ 。

如果 G 为有穷集合, 则称群 G 为有限群, 否则称群 G 为无限群。 G 中元素个数称为群 G 的阶, 记为 $|G|$ 。

若有 $a \in G$ 使 $G = \{a^n | n = 0, \pm 1, \pm 2, \dots\}$, 则称 G 为循环群, a 为 G 的一个生成元, 记为 $G = \langle a \rangle$ 。

设 H 为 G 的非空子集。若对任意 $a, b \in H$ 有 $ab^{-1} \in H$, 则称 H 为 G 的一个子群, 记为 $H \leq G$ 。 $\{e\}$ 为 G 的子群, 称为单位子群。若 $H \leq G$ 且 $H \neq G$, 则称 H 为 G 的真子群。

设 $H \leq G$ 。对任意 $a \in G$, 称 $aH = \{ah | h \in H\}$ 和 $Ha = \{ha | h \in H\}$ 为 G 关于 H 的左陪集和右陪集。 G 关于 H 的不同左 (右) 陪集都互不相交且 $\{aH | a \in G\} = \{Ha | a \in G\}$, 并称 G 关于 H 的不同左陪集个数为 H 对 G 的指数, 记为 $[G:H]$ 。

设 $H \leq G$, 若对任意 $a \in G$ 有 $aH = Ha$, 则称 H 为

G 的一个正规子群, 记为 $H \trianglelefteq G$ 。这时, 对任意 $a, b \in G$ 必有

$$aH \circ bH \trianglelefteq \{ah * bh' | h, h' \in H\} = abH$$

因此 $\{aH | a \in G\}$ 关于上面定义的左陪集乘法成群, 称为 G 关于 H 的商群, 记为 G/H 。

如果群 G (幺半群或半群) 的乘法 $*$ 满足交换律, 即对任意 $a, b \in G$ 有 $ab = ba$, 则称 G 为交换的。交换群又称阿贝尔群。这时, 常将 $*$ 改作 $+$, 并相应地把“乘法”、“积”、“幺元 e ”、“逆元 a^{-1} ”和“商 ab^{-1} ”改称“加法”、“和”、“零元 0 ”、“负元 $-a$ ”和“差 $a - b$ ”。

循环群都是交换群。

若 A 为非空集合, 则 $\{f | f: A \rightarrow A \text{ 为双射}\}$ 关于函数合成构成成群, 称为 A 上的变换群。 $A = \{1, 2, \dots, n\}$ ($n \geq 1$) 上的变换群称为 n 次对称群, 用 S_n 表示。 S_n 的子群称为 n 次置换群。变换群、对称群和置换群一般不是交换群。

设 G 和 G^* 为两个群。若 $f: G \rightarrow G^*$ 对任意 $a, b \in G$ 有 $f(ab) = f(a)f(b)$, 则称 f 为一个群同态, 并称 $\ker(f) = \{a \in G | f(a) \text{ 为 } G^* \text{ 的幺元}\}$ 为 f 的同态核。若群同态 f 为双射, 则称 f 为群同构, 并称 G 与 G^* 同构, 记为 $G \cong G^*$ 。

每个群都与某个变换群同构, 每个有限群都与某个置换群同构。

设 G 和 G^* 为两个群。若 $f: G \rightarrow G^*$ 为群同态, 则 $\ker(f)$ 为 G 的正规子群, 而且当 $f(G) = G^*$ 时有 $G/\ker(f) \cong G^*$ (群同态定理)。

半群、幺半群和群已经是数学及其应用中最基本的概念之一, 已渗透到现代数学的许多分支并起着重要作用, 还形成了一些新学科。此外, 它们在理论物理、结晶学, 特别是计算机科学中, 诸如形式语言、自动机理论和编码理论等, 都有重要应用。

参考文献

1. Hangerford T W 著. 代数学. 冯克勤译. 长沙: 湖南教育出版社, 1985
2. 张远达. 有限群构造, 上, 下. 北京: 科学出版社, 1984
(王兵山 王水汀)

qunjian

群件 (groupware) 为用户提供访问某共享环境的界面, 以支持他们去实现共同总目标或完成共

同总任务的计算机应用系统。群件一词最早出现于 1982 年 P. Johnson - Lenz 和 T. Johnson - Lenz 所撰写的一篇名为“Groupware: The process and impacts of design choices”的论文中。在那里, 群件指的是考虑了社会群体协作过程的以计算机为基础的系统。一般一个群体会在 5 个不同的层次上进行协作, 即

(1) 数据通信 指数据的传送与交换。它是协作赖以完成的基础。

(2) 信息通信 指通过各种媒体或资源以匿名方式进行信息交流与共享。

(3) 协调 其主要目的是保证个体动作之间的同步及其与整个协作过程之间的一致。

(4) 合作 指多个独立的协作成员由于某种工作关系而在一起参与同一过程、执行某种行动。合作是否成功取决于成员的共同理解和共享资源。合作结果归属协作全体而非某特定个体, 但并不排除个体可有其自己的目标。

(5) 协同 最高层次的协作。在这一层次上, 协作体的共同目标完全取代个体目标, 协作成员之间的竞争是最少的, 协作体是以整体而不是以个体核定的。协同过程中往往要共享知识与资源, 以达成共识, 做出具有高可信度与可靠性的共同决策。

一个群件或群件系统就要具有支持上述不同层次的协作的能力, 以支持人们去实现共同总目标或完成共同总任务。因此, 群件系统应具有下列 3 个特征:

(1) 共同总目标或共同总任务 这是人们进行协作的基础。

(2) 共享环境 人们之间的协作是以相互之间的信息与数据交换为基础的。群件系统将为此提供一个共享的信息空间, 也被称之为共享环境。共享环境的作用, 就是让协作用户能够方便地按他们自己的需要去访问、处理这些数据对象, 并将相应的处理结果借助于某种方式及时让其他感兴趣的协作者感知到。

(3) 计算机协作应用系统 群件系统是计算机协作应用系统而不是其他类型的系统, 即用户直接使用它就可以进行某种类型的协作应用, 而无需再进行其他进一步的开发。这将成为未来计算机软件发展的一个重要方向。

上述特点就是群件系统区别于其他系统的显著特征。不同的群件系统在所支持的协作的类型、实现这种支持所用的技术手段等方面可能会有显著的

差别,但只要用这些特征去加以判断,即可以方便地决定一个系统是否属于群件系统。

群件系统分类

人类协作方式的多样性及计算机在各种不同领域的广泛应用使得我们几乎不可能去开发一个能够支持各种类型协作的通用群件系统。经常更多的情况是去开发一些能够支持某种特定类型或适用于某

特定领域协同工作的系统。由此而导致大量的群件系统被开发出来并被投入实际应用。有些系统还逐步成为商品化软件。通常人们按照协作发生的时间与空间分布关系对各种不同的协作方式进行分类(参见计算机支持的协同工作)。同样,对群件系统也可按其所支持的主要协作方式而加以分类,如图1所示。

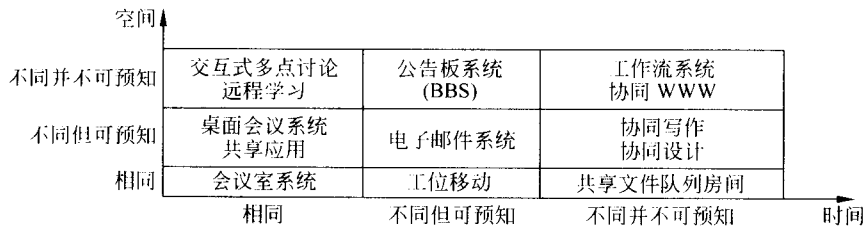


图1 群件系统分类

群件系统设计

群件系统所追求的目标不仅仅是如何用计算机去支持人们完成他自己的那部分工作,它关注的重点更多地在于如何用连成网络的计算机系统去支持多个人构成的群组进行随时随地的交流与协作。这种系统目标及应用环境上的改变使得在设计群件系统时会遇到在设计其他系统(如单用户系统或传统意义上的信息管理系统等)时不曾涉及的一些问题,其中有一些是技术上的,有一些则是非技术上的。包括:

(1) 体系结构 群件系统的设计首先一个问题就是需要根据所要支持的协作的性质、系统赖以运行的软、硬件环境等因素选择一种合适的体系结构。各种不同的体系结构各有其优、缺点,各自所适用的场合也不一样。被现有的群件系统广泛采用的体系结构有集中结构、复制结构和两者的混合结构。

(2) 群组用户界面 群件的用户界面与其他类型的用户界面的一个最重要的差别是:它可能会同时受多个用户的共同控制,需要兼顾对个人活动及群组活动的支持,需要综合使用各种可能的技术手段创造一种尽量自然的交互方式,高度的灵活性、无所不在的访问能力和增强对协作环境的描述能力,同时考虑到同步方式和异步方式的差异性(参见计算机支持的协同工作)。一般而言,实时群件系统大都采用了一种被称为是“你所见即我所见”(WYSIWIS)风格的多媒体用户界面。在异步的群件系统中一般使用的是一种类似于消息系统或电子

邮件系统的用户界面,一般大量使用结构化的、半结构化的或非结构化的消息在协作者之间传递信息以协调他们的活动。等待用户处理的消息(也即任务)将以列表的形式给用户显示出来,用户可以选择其中的某一条进行处理,并在处理完之后将其提交到系统中。

(3) 并发控制 群件系统有其特殊的应用环境,例如:群件系统的主要目标是给多个用户提供一种能够方便地进行交互的共享空间;消除协作用户之间的距离感即广域分布性;为了满足对于响应时间及可靠性的要求,实时群件系统通常采用一种全复制式的体系结构(即将共享对象在各协作者处分别复制一份),等等。因此,对于群件系统并发控制总的要求是:在全复制的体系结构下,在保证系统具有比较好的响应时间和通知时间的前提下,保证复制在各站点处的数据对象的一致性,实现用户之间方便、灵活、自由的交互。

参考文献

1. 史美林. 计算机支持的协同工作: 理论和应用. 北京: 电子工业出版社, 2000
2. Proceedings of the conference on computer supported cooperative work: CSCW'86, 88, 90, 92, 94, 96, 98, 2000
3. Proceedings of the european conference on computer supported cooperative work: E-CSCW'89, 91, 93, 95, 97, 99 (史美林)

R

raojie

绕接 (wire-wrap connection) 用绕接工具把单股导线剥去外皮露出导体,并将导体紧绕在接线端子上,以达到电气连接的工艺过程。在导线环绕拉力作用下,导线与端子的锐利棱边接触,在接触处产生约 $1.03 \times 10^9 \text{ Pa}$ ($10\,500 \text{ kgf/cm}^2$) 的强大压力,使导线和金属端子上的氧化层被破坏,形成清洁的金属接触,并有良好的气密性,能避免再受到氧化和腐蚀,并使连接获得优良的导电性能。试验表明,绕接点可以保证 40 年以上的可靠连接。

绕接技术是由贝尔电话实验室发明的,1952 年首先用在纵横制自动电话交换机上,不但提高了机器的可靠性,节省了各种工艺材料,还缩短了连线的施工时间。

绕接只适用于单股导线,或采用镀锡工艺将多股聚合成一股的导线,绕接线的直径一般从 $0.25 \sim 1.3 \text{ mm}$ 。适合绕接的端子必须具有两个以上的锐角,一般有六种形状:方形、U 形、V 形、滚花扁平形、滚花菱形、双三角形等。在计算机底板接线中常用的是方形。

绕接有两种形式。图 1(a)所示为标准绕接,即常规型绕接。它是将剥去绝缘层的导线,以紧密螺旋方式牢固地缠绕在端子上,一般适用于直径 0.5 mm 以上的导线。图 1(b)为防震型绕接,即除金属部分外,还将一部分未剥绝缘层的导线也缠绕在端子上,绝缘层部分至少绕有三个以上的棱角,以达到良好的防震性。

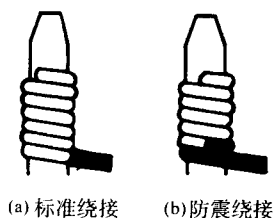


图 1 绕接点的种类

绕接在 20 世纪 70 年代已推广应用于计算机、

通信设备、家用电器以及各种电气控制装置中,绕接用的工艺设备、工具、线材等已不断完善配套,并制订了绕接线材、端子、绕接质量检验等各种标准。绕接可用微型计算机编程全自动进行,定位速度达 10 in/s ($1 \text{ in} = 25.4 \text{ mm}$)。(顾本斗)

resheji

热设计 (thermal management, thermal design) 对计算机系统电子元件、电路模块、设备等的工作温度和温度梯度加以经济、合理、有效的控制,并考虑温度和噪声对人的影响而进行的设计。

电子元器件的性能与温度有关,其故障率随温度的升高成指数关系增加。近年来,大规模和超大规模集成电路的迅速发展,使计算机的热通量(单位为 W/cm^2)和热密度(单位为 W/cm^3)越来越高;同时为了提高可靠性,元器件的工作温度又应有所降低;这些都对热设计提出更高的要求。热设计的目的是将众多元器件产生的热量有效地转移出去,以提高系统的可靠性。在热量转移的路径中存在热阻。计算机中的热阻一般可分为 3 级:

(1) 元件级热阻(即内热阻) 元器件结点与元器件表面(或散热器表面)上参考点间的热阻。系统热设计人员通常无法改变元器件的内热阻。

(2) 插件级热阻 元器件表面参考点与对流层中某指定点间的热阻。

(3) 系统级热阻 插件与最终换热器之间的热阻。最终换热器可能是计算机房的空调系统或冷水机组。

为了达到计算机系统正常工作所要求的温度和温度梯度,需要进行系统的热设计。热设计同时也要考虑到可靠性指标,安装空间限制,维护以及成本等因素。这些因素往往是相互矛盾的,需要经过一系列技术方案的分析比较折衷之后才能确定最佳选择。热设计的重点在于选择冷却形式并进行设计和计算。

计算机的冷却形式有空气自然对流冷却、空气强迫对流冷却、水强迫对流冷却和相变冷却(如碳氟化物自然对流、碳氟化物沸腾)等。图 1 表示各种

冷却形式的适用范围。

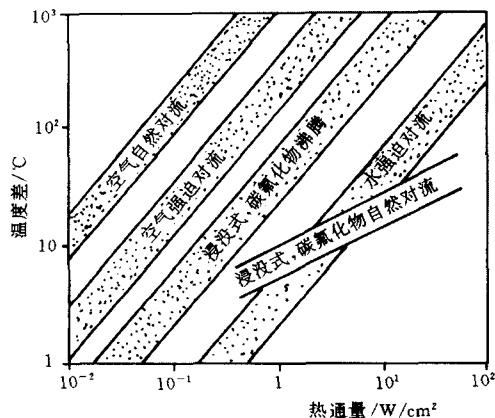


图1 各种冷却形式的适用范围

空气强迫对流冷却 用风机加压使空气流过机柜带走热量。此形式结构简单,维护方便,可靠性高,成本低,目前国内外采用较多。空气强迫对流冷却可分为长、短风路两种。在长风路形式中空气进入机柜后经过几层插件才排出机柜,如图2所示。短风路形式则是空气进入机柜后只经过一层插件即排出机柜,如图3所示。短风路形式机柜内温度梯度小;长风路形式风量大,可提高组装密度。

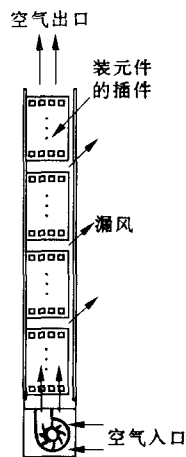


图2 长风路形式

水强迫对流冷却 用水代替空气作冷却介质,换热系数可提高1~2数量级。因水有腐蚀性且绝缘性能差,通常采用间接的水冷方式,即被冷却的元件不与水直接接触,热量经其他介质传导到水,再经对流带走。水强迫对流冷却能有效地降低元器件的外

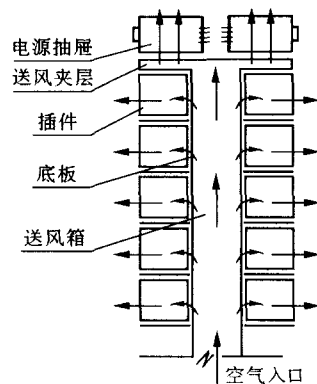


图3 短风路形式

热阻,且具有噪音小,机柜内温度均匀等优点。但它技术较复杂,成本较高,目前多用于大型和巨型机的功率密度高的主机机柜。

间接相变冷却 这种形式采取传导与相变换热相结合的方式。如Cyber 203/205机的陶瓷基片是用弹簧卡压紧在冷却管上(参见图4),冷却管内通氟里昂,氟里昂蒸发将热量带走。这种形式热阻较低,组件的热通量可达 2 W/cm^2 。

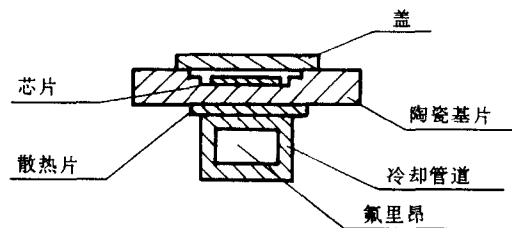


图4 间接相变冷却形式

直接相变冷却 当功率密度进一步提高时,间接相变冷却不能满足要求,就要采取将元器件直接浸没在冷却液(如碳氟化合物)中的这一非常有效的换热方式。直接相变冷却对冷却液除了要求换热效率高之外,还要求介电强度高,化学惰性好。在结构上要求密封严格,因此成本也高。

热电致冷 这是正在研究但已取得实用进展的冷却技术。根据珀耳帖效应,当直流电通过两种不同材料的结点时,结点将吸热或放热。**热电致冷器**的电偶是利用特制的N型和P型半导体材料,用铜连接片焊接而成。单个的电偶冷却能力小,为了增大冷却能力,通常热电致冷器由几十对电偶组成。将热端贴在散热器上,冷端贴在被冷却的元器件上,

能使元器件的温度达到或低于环境温度,温差可达 65°C ,可带走 125 W 的热流量。为了增大温差,可采用多级热电致冷器,4级的热电致冷器温差可达 125°C 。热电致冷器的优点是:改变电流的方向可进行加热或冷却,改变电流的大小可调整换热量,无运动部件,无噪声,在失重或高重力条件下及任何方位均可很好地工作。缺点是体积较大,目前效率仍较低,即输入功率与致冷量之比较高,这种情况需靠发展新的半导体材料得以改善。

热管 热管是利用毛细管原理,采用介质相变来吸热或放热的真空密封传热器件,其传热能力比同样截面的铜要大几十倍,有资料报道可以排除 $47\text{ W}/\text{cm}^2$ 的热通量,在计算机中可用来冷却大功率集成电路。热管的优点是传热能力大,无噪声,但目前价格较高,对真空密封要求很严格(否则很容易失效)。(唐玛琍)

rengong shenjing wangluo

人工神经网络 (artificial neural networks)

由大量处理单元互连组成的非线性、自适应信息处理系统。它是在现代神经科学研究成果的基础上提出的,试图通过模拟大脑神经网络处理、记忆信息的方式进行信息处理。

人工神经网络具有四个基本特性:

(1) 非线性 非线性关系是自然界的普遍特性。大脑的智慧就是一种非线性现象。人工神经元处于激活或抑制两种不同的状态,这种行为在数学上表现为一种非线性关系。具有阈值的神经元构成的网络具有更好的性能,可以提高容错性和存储容量。

(2) 非局域性 一个神经网络通常由多个神经元广泛连接而成。一个系统的整体行为不仅取决于单个神经元的特征,而且可能主要由单元之间的相互作用、相互连接所决定。通过单元之间的大量连接模拟大脑的非局域性。联想记忆是非局域性的典型例子。

(3) 非定常性 人工神经网络具有自适应、自组织、自学习能力。神经网络不但处理的信息可以有各种变化,而且在处理信息的同时,非线性动力系统本身也在不断地变化。经常采用迭代过程描写动力系统的演化过程。

(4) 非凸性 一个系统的演化方向,在一定条件下将取决于某个特定的状态函数,例如能量函数,它的极值相应于系统比较稳定的状态。非凸性是指

这种函数有多个极值,故系统具有多个较稳定的平衡态,这将导致系统演化的多样性。

人工神经网络中,神经元处理单元可表示不同的对象,例如特征、字母、概念,或是一些有意义的抽象模式。网络中处理单元的类型分为三类:输入单元、输出单元和隐单元。输入单元接受外部世界的信号和数据;输出单元实现系统处理结果的输出;隐单元是处在输入与输出单元之间,不能由系统外部观察的单元。神经元间的连接权值反映了单元间的连接强度,信息的表示和处理体现在网络处理单元的连接关系之中。人工神经网络是一种非程序化、适应性、大脑风格的信息处理,其本质是通过网络的变换和动力学行为得到一种并行分布式的信息处理功能,并在不同程度和层次上模仿人脑神经系统的信息处理功能。它是涉及神经科学、思维科学、人工智能、计算机科学等多个领域的交叉学科。

人工神经网络是并行分布式系统,采用了与传统人工智能和信息处理技术完全不同的机理,克服了传统的基于逻辑符号的人工智能在处理直觉、非结构化信息方面的缺陷,具有自适应、自组织和实时学习的特点。

历史沿革 1943年,心理学家 W. S. McCulloch 和数理逻辑学家 W. Pitts 建立了神经网络的数学模型,称为 MP 模型。他们通过 MP 模型提出了神经元的形式化数学描述和网络结构方法,证明了单个神经元能执行逻辑功能,从而开创了人工神经网络研究的时代。1949年,心理学家提出了突触联系强度可变的设想。20世纪60年代,人工神经网络得到了进一步的发展,更完善的神经网络模型被提出,其中包括感知器和自适应线性元件等。M. Minsky 等仔细分析了以感知器为代表的神经网络系统的功能及局限后,于1969年出版了《Perceptron》一书,指出感知器不能解决高阶谓词问题。他们的论点极大地影响了神经网络的研究,加之当时串行计算机和人工智能所取得的成就,掩盖了发展新型计算机和人工智能新途径的必要性和迫切性,使人工神经网络的研究处于低潮。在此期间,一些人工神经网络的研究者仍然致力于这一研究,提出了适应谐振理论(ART网)、自组织映射、认知机网络,同时进行了神经网络数学理论的研究。以上研究为神经网络的研究和发展奠定了基础。1982年,美国加州工学院物理学家 J. J. Hopfield 提出了 Hopfield 神经网络模型,引入了“计算能量”概念,给出了网络稳定性判断。1984年,他又提出了连续时间 Hopfield 神经网络

络模型,为神经计算机的研究做了开拓性的工作,开创了神经网络用于联想记忆和优化计算的新途径,有力地推动了神经网络的研究。1985年,又有学者提出了玻耳兹曼机模型,在学习采用统计热力学模拟退火技术,保证整个系统趋于全局稳定点。1986年进行认知微观结构的研究,提出了并行分布处理的理论。人工神经网络的研究受到了各个发达国家的重视,美国国会通过决议将1990年1月5日开始的10年定为“脑的十年”,国际研究组织号召它的成员国将“脑的十年”变为全球行动。在日本的“真实世界计算(RWC)”项目中,人工神经网络的研究成了一个重要的组成部分。

基本内容 人工神经网络模型主要考虑网络连接的拓扑结构、神经元的特征、学习规则等。目前,已有近40种神经网络模型,其中有反传网络、感知器、自组织映射、霍普菲尔特网络、玻耳兹曼机、适应谐振理论等。根据连接的拓扑结构,神经网络模型可以分为:

(1) 前向网络 网络中各神经元接受前一级的输入,并输出到下一级,网络中没有反馈,可以用一个有向无环路图表示。这种网络实现信号从输入空间到输出空间的变换,它的信息处理能力来自于简单非线性函数的多次复合。网络结构简单,易于实现。反传网络是一种典型的前向网络。

(2) 反馈网络 网络内神经元间有反馈,可以用一个无向的完备图表示。这种神经网络的信息处理是状态的变换,可以用动力学系统理论处理。系统的稳定性与联想记忆功能有密切关系。霍普菲尔特网络、玻耳兹曼机均属于这种类型。

学习是人工神经网络研究的一个重要内容,它的适应性是通过学习实现的。根据环境的变化,对权值进行调整,改善系统的行为。由Hebb提出的Hebb学习规则为神经网络的学习算法奠定了基础。Hebb规则认为学习过程最终发生在神经元之间的突触部位,突触的联系强度随着突触前后神经元的活动而变化。在此基础上,人们提出了各种学习规则和算法,以适应不同网络模型的需要。有效的学习算法,使得神经网络能够通过连接权值的调整,构造客观世界的内在表示,形成具有特色的信息处理方法,信息存储和处理体现在网络的连接中。

根据学习环境不同,神经网络的学习方式可分为监督学习和非监督学习。在监督学习中,将训练样本的数据加到网络输入端,同时将相应的期望输出与网络输出相比较,得到误差信号,以此控制权值

连接强度的调整,经多次训练后收敛到一个确定的权值。当样本情况发生变化时,经学习可以修改权值以适应新的环境。使用监督学习的神经网络模型有反传网络、感知器等。非监督学习时,事先不给定标准样本,直接将网络置于环境之中,学习阶段与工作阶段成为一体。此时,学习规律的变化服从连接权值的演变方程。非监督学习最简单的例子是Hebb学习规则。竞争学习规则是一个更复杂的非监督学习的例子,它是根据已建立的聚类进行权值调整。自组织映射、适应谐振理论网络等都是与竞争学习有关的典型模型。

研究神经网络的非线性动力学性质,主要采用动力学系统理论、非线性规划理论和统计理论,来分析神经网络的演化过程和吸引子的性质,探索神经网络的协同行为和集体计算功能,了解神经信息处理的机制。为了探讨神经网络在整体性和模糊性方面处理信息的可能,混沌理论的概念和方法将会发挥作用。混沌是一个相当难以精确定义的数学概念。一般而言,“混沌”是指由确定性方程描述的力学系统中表现出的非确定性行为,或称之为确定的随机性。“确定性”是因为它由内在的原因而不是外来的噪声或干扰所产生,而“随机性”是指其不规则的、不能预测的行为,只可能用统计的方法描述。混沌动力学系统的主要特征是其状态对初始条件的灵敏依赖性,混沌反映其内在的随机性。混沌理论是指描述具有混沌行为的非线性动力学系统的基本理论、概念、方法,它把动力学系统的复杂行为理解为其自身与其在同外界进行物质、能量和信息交换过程中内在的有结构的行为,而不是外来的和偶然的行爲,混沌状态是一种定态。混沌动力学系统的定态包括:静止、平稳量、周期性、准周期性和混沌解。混沌轨线是整体上稳定与局部不稳定相结合的结果,称之为奇异吸引子。一个奇异吸引子有如下一些特征:①奇异吸引子是一个吸引子,但它既不是不动点,也不是周期解;②奇异吸引子是不可分割的,即不能分为两个以及两个以上的吸引子;③它对初始值十分敏感,不同的初始值会导致极不相同的行为。

发展趋势 人工神经网络特有的非线性适应性信息处理能力,克服了传统人工智能方法对于直觉,如模式、语音识别、非结构化信息处理方面的缺陷,使之在神经专家系统、模式识别、智能控制、组合优化、预测等领域得到成功应用。人工神经网络与其他传统方法相结合,将推动人工智能和信息处理技

术不断发展。近年来,人工神经网络正向模拟人类认知的道路上更加深入发展,与模糊系统、遗传算法、进化机制等结合,形成计算智能,成为人工智能的一个重要方向,将在实际应用中得到发展。将信息几何应用于人工神经网络的研究,为人工神经网络的理论研究开辟了新的途径。神经计算机的研究发展很快,已有产品进入市场。光电结合的神经计算机为人工神经网络的发展提供了良好条件。

参考文献

1. Rumelhart D E, McClelland J L. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press, 1986
2. 史忠植. 神经计算. 北京: 电子工业出版社, 1993
(史忠植 张建)

rengong shenjing wangluo zai moshi shi-bie zhong de yingyong

人工神经网络在模式识别中的应用 (application of artificial neural networks to pattern recognition) 模式识别是人工神经网络应

用的一个重要方面,使用的网络形式有很多种。最简单的单层网络就是早期的感知机,当样本是线性可分时,用感知机学习算法在有限步内可以收敛到解;也可以用使均方误差达最小的方法去训练感知机,这样即使样本不是线性可分时,它也可收敛于使MSE最小的解,此时它可等价于模式识别中常用的Fisher线性判别函数。

含有隐层的多层前向网络(图1,其中 W 为权重)可以构造出复杂的分界面,所以它是模式识别中最常用的网络形式。由于只含一个隐层的网络可以构造出足够复杂的分界面,所以目前用得最多。多层网络结构与分类区的关系见表1。

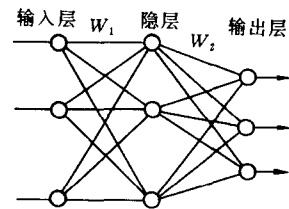


图1 多层前向网络

表1 多层网络结构与分类区域的关系

网络结构	分类区域类型	对异或(XOR)分类	对咬合状区域的分类	分类区域一般形状
无隐层 	由超平面分成两个区域			
单隐层 	开凸区域或闭凸区域			
两个隐层 	任意形状(但与节点数之间的关系复杂)			

由表可见,无隐层的网络只能区分可由超平面分开的类区;单隐层的网络可以形成凸区域;含两个隐层的网络则可形成任意复杂的类区(包括非连通的类区)。

在理想情况下(训练样本和网络规模都不受限制),多层前向网络的输出可任意逼近样本的后验概率分布,所以它可以趋近最佳分类器(Bayes 分类

器)。当训练样本有限时,应当合理选择网络规模(主要是隐层数及各隐层单元数),以使网络有较好的学习效果和推广能力。

多层前向网络的隐单元的作用函数有许多形式,常用的是S形函数(或称Sigmoid函数),其形式为

$$f(x) = \frac{1}{1 + e^{-x}}$$

也可以采用径向基函数(RBF),它是一种中心对称的函数,例如高斯函数:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-c)^2}{2\sigma^2}\right)$$

它有两个参数,中心位置 c 和宽度 σ ,如图2所示,此时网络的作用相当于用 Parzen 法恢复概率密度。因而是一种非参数法的分类器。

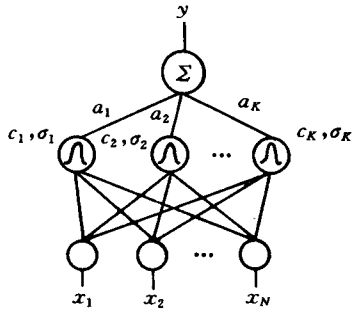


图2 用径向基函数的多层前向网络

另一种用于模式识别的神经网络是**概率神经网络**,它是一个有两层隐单元的多层前向网络,如图3,其中第一隐层(模式单元)的作用函数是高斯函数,第二隐层用求和函数,输出单元是求和并判别,适当选择各连接权的值,可使 $f_A(X)$, $f_B(X)$ 分别代表两类(A类和B类)的用方差为 σ 的正态核函数估计的分布密度,再经加权组合进行判别分类。控制高斯函数方差 σ 的大小,可获得不同类型的分类器, σ 为某固定常数时,相当于恢复分布的分类器, σ 趋近无限大时,相当于分段线性分类器, σ 为零时相当于近邻分类器。

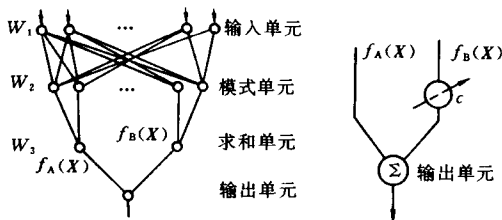


图3 概率神经网络

用多层前向网络也可以实现树分类器,图4表示一个二值决策树对应的多层前向网络,第一隐层(区分层)的作用是将特征空间划分成不同的开

域,第二隐层(与层)用于形成凸区域,输出层是或层,用于联合各凸区以形成不同类区。

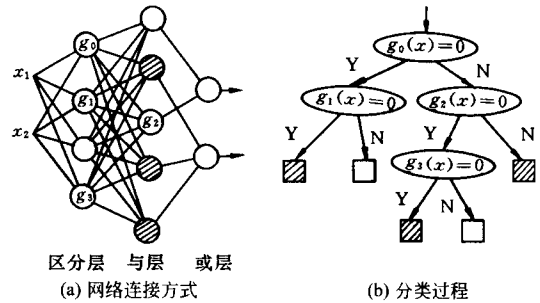


图4 用神经网络实现二值决策树

除了上面所讲的用监督学习方法构造分类器外,神经网络也被用于**非监督学习**,即聚类分析;其中最常用的是 T. Kohonen 提出的自组织映射和相应的学习算法,它可完成自动聚类的功能。

除了做分类器外,人工神经网络也可用于特征压缩。对于一个具有 N 个输入的单个神经元,如果输入 $X = (x_1, x_2, \dots, x_N)^T$ 是均值 $E[X] = 0$,协方差阵 $R = E[XX^T]$ 的随机向量,设输出为 y ,权系数 $W = (w_1, w_2, \dots, w_N)^T$,可以证明如果权向量按下式修改:

$$\Delta W = \mu y [X - W^T X W]$$

其中 μ 为步长,则 W 将收敛于对应 R 最大特征值的特征向量,据这一方法可以实现主成分分析,从而可用于特征的压缩。

上面讲的方法都是基于神经网络的**映射作用**,用于分类的知识通过向样本学习隐含于网络的映射函数中。还有一种基于知识的分类器,它显式利用知识和推理规则把样本分类(相当于一个用于模式识别的专家系统),它也可以用神经网络实现,这种方法的缺点是学习过程复杂。如果把两种方法结合起来构成用神经网络组成的混合型分类器,则既可以方便地利用知识又具有很好的学习能力。

模糊推理与神经网络结合具有一系列优点,它

更接近人类的思维过程。在模式识别方面,用神经网络可实现基于模糊推理的分类器和模糊聚类算法。当然也可以实现如前所述的基于模糊知识和映射的混合型模式识别系统。

从应用领域看,人工神经网络多用于语音识别和文字识别,其他如目标识别、生物医学图像及信号中的有关问题,遥感图像分类、故障诊断等领域也都有应用。

参考文献

1. Zurada J M. Introduction to Artificial Systems. West Publishing Company, 1992
2. 阎平凡,黄端旭. 人工神经网络——模型,分析与应用. 合肥:安徽教育出版社,1993
3. Lippmann R P. Pattern Classification Using Neural Networks. IEEE. Communication Magazine, 1989, 27(11): 47~64 (阎平凡)

rengong shengming

人工生命 (artificial life) 采用计算机和其他精密机电装置,表现自然生命系统行为特点的仿真系统与技术或模型系统与技术。

人工生命的研究可以追溯到 20 世纪中叶计算机专家图灵关于生物的胚胎发育的数学思想以及冯·诺依曼关于细胞自动机的思想。他们的这些思想虽然对人工生命意义重大,但由于当时的计算机速度的限制,并没有立刻引起人们的关注。到了 70 年代和 80 年代,随着计算机速度的大幅度提高以及个人计算机的普及,人们开始重新审视他们的这些工作。在有关“生命游戏”研究的基础上发现,处于“混沌的边缘”的细胞自动机既有足够的稳定性储存信息,又有足够的流动性来传递信息。当把这种规律与生命和智能联系起来就会认识到,生命或者智能很可能就起源于“混沌的边缘”。如果在计算机中建立起产生“混沌的边缘”的一定规则,那么,从这些规则中就有可能浮现出生命来。由于这种生命不同于地球上以碳为基础的生命,因此称为“人工生命”。

人工生命的研究可以采取下列策略:

(1) 采用以计算机等信息处理机器为中心的硬件生成生命行为。一般有两种方法:一种是采用已有的信息处理机器和执行装置,实现具有人工生命行为的系统。另一种是用生物器件构造生命系统。这些都称为生物计算机,是一种向人工生命接近的方法(参见生物计算)。

(2) 用计算机仿真,研究开发显示生命体特征行为的模型软件。简单地说,神经网络系统和遗传算法等,都是采用信息数学模型,模拟人工生命的生成。

(3) 基于工作原理,利用计算机仿真生成生命体。生命现象的基础是随物理熵的增大而杂乱无章。生成这种现象的原理是混沌的分形,耗散结构,协同反应等。

(4) 通过计算机仿真,分析生命特有的行为生成,建立新的理论。

人工生命研究的基础理论是细胞自动机理论、形态形成理论、混沌理论、遗传理论等。

人工生命是形成新的信息处理体系强大的推动力,并成为研究生物的一个特别有用的工具。人工生命的研究可能将信息科学和生命科学结合起来,形成生命信息科学。在 21 世纪初人工生命的研究将会蓬勃发展,并取得更大进展。(史忠植)

rengong zhineng

人工智能 (artificial intelligence) 研究解释和模拟人类智能、智能行为及其规律的一门学科。其主要任务是建立智能信息处理理论,进而设计可以展现某些近似于人类智能行为的计算系统。它是计算机科学的一个分支,也为某些相关学科如心理学等所关注。

20 世纪 40 年代,在人工智能的萌芽时期。有两种不同的研究智能的途径,一是根据神经心理学的研究,通过为神经活动建立数学模型来表现智能行为,这是智能的结构(微观)研究观点;二是从智能行为的角度来研究智能,而不介意这种行为的产生原因,是智能的行为研究观点。前者的主要研究成果是 1943 年 W. McCulloch 与 Pitts 为神经元建立的数学模型,以及 Wiener 的控制论。后者的有代表性的研究是 Turing 实验,目的是建立一种判别机器具有智能的准则;关心的是智能行为的体现,而不考虑完成智能任务的机器是否与智能生物有相同的组成结构。此外, C. Shannon 研制了第一个计算机下棋程序,证明了计算机进行非数值(符号)处理的能力,同时这个程序使用了搜索方法,这正是人工智能的基本方法之一。

20 世纪 50 年代,人们发现从微观研究智能在技术上存在着难以克服的困难,而心理学所提供的试验数据以及逻辑学的进展所提供的模型使行为主义的智能观(或称符号机制)较容易在计算机上实

现。1956 年在美国的 Dartmouth 讨论会上, J. McCarthy, H. Simon, A. Newell 与 M. Minsky 等学者提出了“人工智能”这个术语, 并开始了人工智能的研究。

20 世纪 60 年代, 人工智能研究出现了高潮。J. McCarthy 发表的符号处理语言原型, 提供了用计算机程序设计作为人工智能符号处理的基础。1960 年 H. Simon 夫妇做了一个有趣的心理学实验, 表明人类的问题求解是一个搜索的过程, 其效率取决于启发式函数。在此基础上, A. Newell, J. Shaw 与 H. Simon 研制了通用问题求解系统 GPS。10 年后 J. Nilsson 提出了启发式搜索算法 A^* 。1965 年 J. Robinson 在研究机器定理证明过程中, 提出了归结原理, 并证明了归结法的完备性。这项研究推动了自动推理的进展。1969 年 M. Minsky 和 S. Papert 合写了《感知机》, 这本书指出了感知机算法存在的问题。

在上述一系列研究的基础上, 出现了基于知识的应用系统研究。E. Feigenbaum 提出了第一个专家系统 DENDRAL, 用于分析化学分子结构。到 70 年代 E. Feigenbaum 提出知识工程, 提倡开展知识工程的研究, 人们逐步认识到领域专家通过实践积累起来的知识的重要性, 并总结出建造专家系统及开发环境的一系列原则。这期间出现了成千上万的专家系统, 涉及到上百个应用领域, 如医学、地质学、分子生物学、化学分析、自动程序设计、计算机辅助教育和智能控制等。这些成果使人们看到了人工智能研究的实际意义, 推动了整个人工智能的进一步研究。专家系统的应用已深入到各个领域, 带来了明显的经济效益和社会效益。在广泛的应用过程中, 人们也发现专家系统获取专家知识的困难性以及系统本身的脆弱性。

20 世纪 80 年代, Hopfield 和 D. Rumelhart 等人对神经网络的研究, 解决了 M. Minsky 所提出的问题, 出现了研究人工神经网络的热潮, 通常称这类研究为连接机制。符号机制与连接机制采用了完全不同的表示方法, 两者相互结合、互为补充, 推动了人工智能研究的进展。80 年代后期, 人工智能引入其他学科的研究成果, 出现了一些引人注目的新方法。例如, H. Holland 提出的基于自然选择的遗传进化模型, 使用类似基因结构的表示法来描述问题, 利用类似生物遗传的方法来发现新的规则。在人工智能的研究和应用方面, 我国学者也做出了令人瞩目的成果。如吴文俊提出的平面几何定理机器证明, 解决了大量相当困难的平面几何证明问题,

受到国际学术界的重视。

人工智能学科研究的主要内容包括: 知识表示, 自动推理和搜索方法, 机器学习和知识获取, 知识处理系统, 自然语言理解, 计算机视觉, 智能机器人, 自动程序设计(参见软件自动化方法)等方面。

知识表示是人工智能的基本问题之一, 推理和搜索都与表示方法密切相关。常用的知识表示方法有: 逻辑表示法、产生式表示法、语义网络表示法和框架表示法等, 近年来使用图像、图形直接表示知识并参加推理的直接表示法已引起人们的关注。常识知识是人工智能研究的重要问题, 常识指人们直觉的、日常使用的那些非专业性知识, 它们不一定在任何情况下都正确, 但对问题的求解常起决定性的作用。例如“鸟会飞”是常识知识, 大多数鸟都会飞, 但也有众多例外, 如“鸵鸟不会飞”。如何表达和使用常识, 自然为人们所关注, 已提出多种方法, 如非单调推理、定性推理就是从不同角度来表达常识和处理常识的。

问题求解中的自动推理是知识的使用过程, 由于有多种知识表示方法, 相应地有多种推理方法。推理过程一般可分为演绎推理和非演绎推理。谓词逻辑是演绎推理的基础。结构化表示下的继承性推理是非演绎性的。由于知识处理的需要, 近几年来提出了多种非演绎的推理方法, 如连接机制推理、类比推理、基于示例的推理、反绎推理和受限推理等。

搜索是人工智能的一种问题求解方法, 搜索策略决定着问题求解的一个推理步中知识被使用的优先关系。可分为无信息引导的盲目搜索和利用经验知识引导的启发式搜索。启发式知识常由启发式函数来表达, 启发式知识利用得越充分, 求解问题的搜索空间就越小。典型的启发式搜索方法有 A^* 、 AO^* 算法等。近几年搜索方法研究开始注意那些具有百万节点的超大规模的搜索问题。

机器学习是人工智能的另一重要课题。机器学习是指在一定的知识表示意义下获取新知识的过程, 按照学习机制的不同, 主要有归纳学习、分析学习、连接机制学习和遗传学习等。

知识处理系统主要由知识库和推理机组成。知识库存储系统所需要的知识, 当知识量较大而又有多种表示方法时, 知识的合理组织与管理是重要的; 推理机在问题求解时, 规定使用知识的基本方法和策略, 推理过程中为记录结果或通信需设数据库或采用黑板机制。如果在知识库中存储的是某一领域(如医疗诊断)的专家知识, 则这样的知识系统称为

专家系统。为适应复杂的问题求解的需要,单一的专家系统向多主体的分布式人工智能系统发展,这时知识共享、主体间的协作、矛盾的出现和处理将是研究的关键问题。

人与机器进行自然的对话,利用能为计算机所接受的自然语言描述现实世界问题,一直是人工智能的研究目标之一。所谓机器理解了某种自然语言,抽象的标准是指机器能通过以该自然语言形式进行的 Turing 实验,具体的标准则随人们对它提出的功能要求而异。例如要求机器能正确地回答提出的问题;能产生输入文本的摘要;能用不同的词语和句型来复述输入文本或能把一种语言翻译成另一种语言等等。机器如能达到上述中的一种要求就将产生广泛的应用。自然语言的理解过程包括语法分析、语义分析和语用分析,已研制的一些比较成功的自然语言理解系统都只能处理自然语言的子集,通常是句法受限或语义受限或领域受限。其中,人机接口和机器翻译系统已有商品,但要让机器像人那样自如地运用自然语言,仍是长远而艰巨的任务。近期,自然语言理解的研究朝实用化、工程化发展,建立在大量语料上的基于语料库的自然语言理解已引起人们的重视。

制造具有某种智能的机器人是工业上和军事上的需要。机器人的研究涉及机械、电子、控制及计算机的诸多方面。从人工智能角度研究机器人,主要涉及表示技术和自动推理技术以及规划方法和感知技术等。目前智能机器人研究更加注意面向现实来解决实际问题。

人工智能的研究已有三十多年的历史,发展是曲折的,在物理符号机制和基于启发式求解的方法指导下,在专家系统、机器翻译、机器视觉和问题求解等方面的研究已有实际应用。近年来对人工神经网络的知识表示、常识推理、机器学习和分布式人工智能等基础性研究也取得了可喜的进展。区别于符号机制和连接机制,有人还提出了无需表示、无需概念的智能观。对逻辑在人工智能中的作用、知识与概念化、认知与学习、认知与感知、计算智能与人工智能的关系等问题开展了有益的辩论。多学科交叉、人机一体化等观点也影响着人工智能的研究。

参考文献

1. Feigenbaum E A. The Handbook of Artificial Intelligence. Vol. I ~ IV. Addison -Wesley Publishing Company, 1981 ~ 1989
2. Minsky M, Papert S. Perceptron (Expanded-

edition). MIT Press, 1988

3. 陆汝钤. 人工智能. 北京: 科学出版社, 1996
(石统一 王珏)

renji jiaohu jishu

人机交互技术 (human-computer interaction technology, or HCI technology)

依据对人与人及人与计算机系统之间的交互等人类行为的理解,使计算机能够以友善、自然和直观的方式与用户进行信息交换或提供服务的技术。

随着计算机技术的发展,如速度提高,存储设备的容量增加等,计算机系统变得更加复杂,其应用也渗透到人类生活的各个方面。相应地,计算机系统的使用人员也由计算机专业人员扩展到非专业人员。简单、易学、易用是用户对这些复杂的计算机系统的基本要求,人机交互技术也应运而生。随着普适计算的发展,用户可以随时拥有多个计算机设备,并得到这些计算机设备提供的主动伺服式服务,从而使人机交互的范围扩展到人们的实际生活中。

人机交互技术的形成是多学科综合作用的结果。除了计算机技术,如多媒体技术、计算机图形学、操作系统、计算机视觉等学科发展的影响以外,人性因素、人类工程学、工业工程、认知心理学、社会心理学等学科的发展,也对人机交互技术的发展起着重要的作用。计算机图形学产生于 20 世纪 60 年代,它的发展形成了多种直接操纵方式的人机交互技术。多媒体计算和智能接口技术的发展又进一步使得有可能以人类惯有的方式进行人机交互。目前操作系统一般都包含有“用户接口管理系统”和“用户接口开发工具”,对系统的个性化使用和应用程序用户接口的开发提供了支持。第二次世界大战时期对设备易操作性的要求引起了人们对人性因素的关注,同样,在人机交互技术的研究中也要考虑用户如何方便地使用计算机设备。人类工程学研究用户工作时生理特性的影响并着重研究环境和系统对用户的影响。工业工程学来源于 20 世纪初提高工业产量的需求,人机交互技术也需要研究如何让计算机应用到更多的工作中以提高生产效率。认知心理学和社会心理学有助于人机交互技术的进步,如研究用户学习某种计算机系统的能力、用户获得的这种能力在其他计算机系统适应程度以及用户在这些系统中的工作效率等。随着个人计算机和工作站的普及,人们对人机接口的要求也越来越高,因而需要对众多的交互接口技术进行规范化,并发展测

试技术以评价各种接口技术的性能。

人机交互技术主要的研究内容有人机执行任务的综合性能,人机之间的交互结构,人类使用计算机的能力(包括学习接口的能力),接口的算法与编程,接口的规范化、设计、实现及其评价。它们的相互关系如图1所示。计算机系统总是要和一定的社会、组织或者工作环境(U1)相联系,在各种环境中用户希望使用的应用类型(U2)也不尽相同,工作的过程意味着用户和计算机相互适应的过程(U3),也就是需要用户学习计算机使用方法,需要系统为用户定制功能等。除了研究使用环境外,从用户方面而言,人机交互技术需要了解用户的信息处理过程

(H1)、用户交流的方式(H2)以及用户的生理特征(H3)等。从计算机的角度来说,人机交互技术包括人机之间的输入输出设备(C1)、人机之间的对话技术(C2)、对话的基本类型(C3);进一步而言,对话技术可以采用多媒体技术、计算机图形技术和虚拟现实技术等(C4),复杂的对话技术需要考虑对话的体系结构(C5)方面的因素,如多用户协同界面、多任务对话模型等。最后,在开发阶段人机交互技术需要考虑集成人机对话的设计方法(D1)、实现技术和开发工具(D2)、对系统的评价方法(D3)以及实例研究(D4)等,开发阶段的每一个过程是相互关联、彼此影响的。

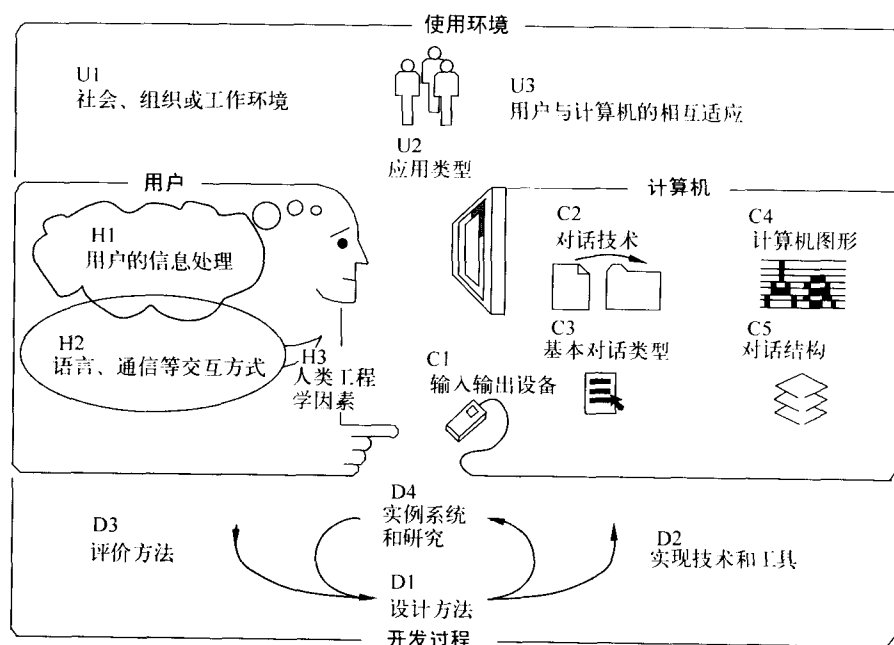


图1 人机交互技术主要的研究内容

(图片引自参考文献3)

参考文献

1. Preece J, Rogers Y, Sharp H, et al. Human-computer Interaction. Addison-Wesley, England, 1994
2. Faulkner C. The Essence of Human-computer Interaction. Prentice Hall, England, 1998
3. Dix A J, Finlay J E, Abowd G D, Beale R. Human-computer Interaction. Second Edition. Prentice Hall Europe, London, 1998 (陶霖密 陈恩义)

renji jiaohu xitong

人机交互系统 (human-computer interaction system) 支持人和计算机系统直接进行交互通信的系统,其主要功能是完成人机之间的信息传递以提高计算机系统的友善性和效率。

分时系统出现后,用户可以在各自的终端上使用计算机,例如,从终端打入命令和输入数据,并从终端上得到计算机系统输出的各种信息,这就是早期的人机交互系统。这种人机交互系统的输入输出设备是具有输入输出功能的控制打字机,相应的软

件是命令解释系统。

随着计算机系统(包括输入输出设备)功能的增强,人机交互系统也有了较大发展。新型输入输出设备,例如,CRT交互终端设备、鼠标器、声音输入和合成设备、图像扫描仪、文字识别设备等的出现,使人机交互系统从用文字进行交互通信发展到可以通过图形、图像、声音等进行交互通信。近来多媒体技术的出现,使人机之间有可能按照人的自然和习惯的方式,通过多种载体或媒体完成信息的交换和处理。

人机交互系统要能很好地实现用户与计算机之间的人机交互,必须考虑三个元素:人的因素、交互设备及实现人机交互的软件。

其中,人的因素是指用户操作模型。要根据用户的类型,固有的特点,设计好的用户操作模型,使人机交互系统满足用户的使用要求。交互设备构成了人机交互系统进行人机对话的基础。它包括数字和字母输入输出设备、图形和图像输入输出设备,以及声音、触感等专用输入输出设备。人机交互软件是人机交互系统的核心,它向用户提供各种交互功能,以满足系统预定的要求。它和所有软件一样可分为系统软件和应用软件。

在系统软件方面,许多分时操作系统均采用命令语言的对话方式向用户提供操作界面,这类操作系统如UNIX,VMS,DOS等。一些高级语言的解释程序(如BASIC,LISP,PROLOG)采用交互式解释执行,而高级语言的编译程序(如Turbo Pascal,Turbo C)则采用编辑、编译、调试等交互式集成程序设计环境,这类语言工具十分便于用户编程和调试。在数据库管理系统中通常也采用交互式数据库查询语言,有的用命令语言(如SQL),也有的用填表方式(如QBE)。在数量众多的软件工具中,已经广泛使用全屏幕正文编辑程序、排错程序,电子表格软件、多窗口系统等交互式软件工具。系统软件中还包括一批可用于辅助生成人机界面的软件工具或环境,应用系统的人机界面可以在它们的基础上开发,或用它们进行辅助开发。多窗口系统、用户界面管理系统(UIMS)等就是这样的工具。交互式图形系统也是这类支持软件之一。

在应用软件方面,交互式人机界面已成为其主要部分之一,并成为衡量应用软件功能强弱的一个重要指标。在人机交互应用系统中,开发人机界面的部分占了相当大的工作量。为了提高人机界面软件的生产率和可重用性,一个重要的发展趋势是

人机界面与应用系统中的功能部分分离出来,并研制自动或辅助生成人机界面的软件工具。由于应用领域的广泛性,不同应用领域的人机交互方式可能迥然不同。

根据人机交互方式的演变,人机交互系统可以大致分为命令语言交互系统,选单驱动交互系统,直接操纵交互系统,多媒体交互系统等。

命令语言交互系统 命令语言是用以请求系统服务并传送系统回答的可执行语言。它由一组命令组成,每一命令又由命令名和若干命令参数构成。命令语言交互系统负责获取用户命令,分析命令的语法、语义结构,实际执行命令赋予的功能,并把系统回答传送给用户,从而让用户通过键入命令来控制 and 操纵计算机系统的运行。

选单驱动交互系统 选单是若干可供用户选择的系统功能表。选单驱动交互系统是通过选择选单项来执行系统功能。选单的内容及组织结构体现了系统的功能及其按层次的分解与组织。选单驱动交互系统鼓励用户在选单引导下遍历系统功能。

直接操纵交互系统 直接操纵交互系统借助图形并通过模拟人对客观世界中实体的操作来完成人机交互并使操作过程和操作效果可视化。例如把表示文件的图符移到表示废纸箱的图符中则比喻对该文件实行删除操作。直接操纵使用户可以用现实生活中认识和处理问题的方式和习惯来解决人机交互问题,所以很受用户欢迎并已广泛用于许多人机交互系统中。

在直接操纵交互系统中,广泛使用了WIMP(窗口、图符、选单、指点器)式图形用户界面技术和面向对象方法。

多媒体交互系统 多媒体交互系统采用文本、图形、图像、声音、视频等多种媒体来表示、储存和处理信息,通过人们耳闻、口述、目睹、手触等多种方式与计算机进行交互,使人机交互更加简单、自然、友善、一致。

人机交互系统的研究内容主要有:

(1) 人机交互系统模型的建立与分析 它研究如何把人的认知模型包含到计算机系统设计中,建立描述人机交互系统的工作原理、组织结构和人机交互活动过程的人机交互模型。常用的人机交互模型有:基于语言描述的结构化分层模型、基于描述时间和逻辑序列的控制模型、基于应用任务的任务分析模型及面向对象模型等。

(2) 人机交互系统工作方式和设计原理 它依

据交互输入输出设备和计算机软件技术的发展,研究适合用户需要的人机交互方式、制定和总结各种交互方式的人机界面的设计原理与准则,来指导界面的设计和用户的选择。

(3) 人机交互系统的设计方法 它研究如何设计和开发界面的屏幕外观形式、确定用户和系统交互方式,并把用户操作处理成对系统功能的控制。常用的设计方法有:使用程序设计语言(如 C, C++)、使用界面工具箱(如 X 窗口下的 X-toolkit 和 Widget 部件集)、使用专门的界面描述语言(如 Motif 标准中的 UIL 语言)、使用直接操纵方式等。

(4) 人机交互系统的评估 它研究如何评价人机交互系统的功能,制定各种评价准则。主要评价性能有:使用的难易程度、学习的难易程度、开发的难易程度、系统的复杂程度、操作速度等。常用的评价方法有:随机分析方法、概率统计方法等。

参考文献

1. 程景云,倪亦泉. 人机界面设计方法与开发工具. 北京: 电子工业出版社, 1994
2. 董士海. 计算机用户界面及其工具. 北京: 科学出版社, 1994 (程景云 倪亦泉)

renlian shibie

人脸识别 (face recognition) 计算机利用人的脸部图像自动识别其身份的过程和技术。计算机对输入的图像或视频, 首先判断其中是否存在人脸, 如果存在, 则进一步给出人脸的位置、大小以及各个主要面部器官的信息, 并依据这些信息, 进一步提取脸像中所包含的身份特征信息, 并与人脸库中的已知人脸进行对比, 从而识别其身份。

脸像是人类视觉中最为普遍的模式, 人脸所反映的视觉信息在人与人的交流、交往中有着重要的作用和意义。人脸识别的研究从 20 世纪 60 年代末开始, 由于其广泛的应用领域, 其研究一直备受瞩目, 依托于图像理解、模式识别、计算机视觉和神经

网络等技术, 其研究取得了很大进展, 并逐渐推向应用领域。

人脸识别领域比较著名的国际研究机构有美国麻省理工学院媒体实验室及人工智能实验室、南加州大学 (USC)、CMU 卡内基-梅隆机器人研究及交互系统实验室、马里兰大学 (UMD) 等。

国内对人脸识别的研究从 20 世纪 90 年代中后期开始。许多研究机构在人脸识别研究领域进行了许多有意义的尝试。在大规模人脸识别领域已经取得突破性进展, 并投入了实际应用。

完整的人脸识别技术至少应包括两个主要环节。首先要在输入的图像或图像序列中找到人脸的位置, 并把人脸从背景中分离出来, 这是人脸检测的主要研究内容。所发展出来的方法主要有基于模板匹配、基于颜色信息、基于特征脸匹配、基于 Ada-Boost 的实时人脸检测方法等。其次, 根据分离后的人脸图像进行特征的提取与识别, 即人脸识别环节。人脸特征主要包括人脸的全局特征、几何特征、各个器官的特征等, 因此为适应这些特征也产生了不同的识别算法。

(1) 基于几何特征和基于模板匹配的方法 这两种方法主要在人脸识别研究的初期使用。基于面部结构几何特征的人脸识别是最直观, 也是最传统的方法, 它对人脸的描述非常紧凑, 但特征提取困难、易受头部姿态变化影响等缺点。基于模板匹配的方法则具有特征提取简单的优点, 其准确度也较高, 但该方法容易受到光照和姿态的影响。因此, 这两种方法通常都需要与其他算法结合, 才能得到比较好的效果。

(2) 特征脸识别方法 基于特征脸 (Eigen-Face) 识别的方法由麻省理工学院 M. Turk 和 A. Pentland 于 1991 年提出, 是目前最流行的人脸识别算法。该方法通过主分量分析 (PCA) 将原始脸像分解成“平均脸”和一系列互相正交的“特征脸”, 并基于此进行识别。图 1 给出了平均脸和特征脸的分解示意。

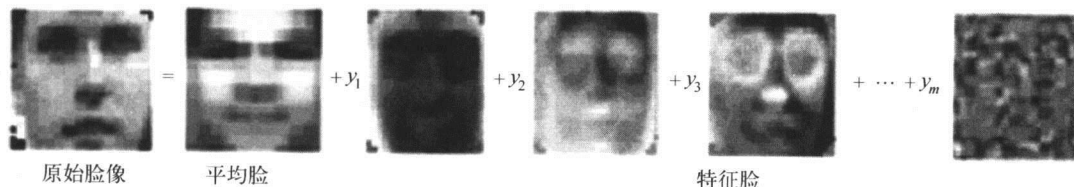


图 1 特征脸分解示意图

该方法具有简单有效的特点,但它对输入的人脸图像的归一化要求较高,性能易受光照和姿态变化的影响。目前研究者对其进行了各种改进和扩展,尝试了基于特征脸和各种后端分类器相结合的方法,如二阶 EigenFace、线性判别分析以及双子空间分析方法等。

(3) 基于融合特征的方法 近年来,研究人员逐渐认识到,一种有效的人脸识别算法,必须能够充分挖掘人脸不同方面的特征信息,也即要有效利用人脸的形状拓扑结构、局部灰度和全局灰度分布等多种特征。因此,出现了融合上述特征信息的一些方法。

基于 Gabor 小波变换和弹性图匹配的方法是一种人脸形状拓扑结构特征和局部灰度特征的算法。另一种融合多种面部特征的方法是基于可变形统计模型的方法,该方法综合利用了人脸形状信息(主动形状模型 ASM)、局部灰度分布特征和全局灰度分布特征(主动外观模型 AAM)。上述方法具有自动提取形状特征或精确定位特征点的功能,因而具有较高的识别精度,但是存在复杂度高、实现复杂的缺点。

(4) 其他方法 此外,局部特征分析(LFA)、隐马尔可夫模型(HMM)、奇异值分解(SVD)、独立元分析(ICA)等方法也在人脸识别中得到了应用。

人脸识别技术具有广泛的应用前景。在各种安全领域,如智能门禁、视频监控等都具有重要的应用价值。在游戏娱乐领域,人脸识别也可以有一些有趣有益的应用,比如能够识别主人身份的智能玩具、具有真实脸像的虚拟游戏等。另外,图像捕捉设备正成为个人计算机的标准外设,因此人脸识别也必将成为未来人机和谐交互的一个重要手段。

必须看到,尽管人脸识别技术已经取得了长足的进展,但是在应用过程中所遇到的一些实际问题,比如光照条件的改变,用户姿态、表情、发型、胡须的变化,真实人脸与照片的辨别,甚至不同摄像机参数的不一致等,都严重影响识别的性能。必须对这些实际问题进行系统的、有针对性的研究。

参考文献

1. Turk M and Pentland A. Eigenfaces for Recognition. J. Cognitive Neuroscience, 1993, 3: 71 ~ 86
2. Martinez A M, Kak A C. PCA versus LDA. IEEE Trans. PAMI, 2001, 23(2): 228 ~ 233

(吴志勇 蔡莲红)

renti donghua

人体动画(human body animation) 利用计算机进行人体造型和行为模拟的计算机动画技术。它综合运用计算机图形学、解剖学、生理学、生物力学、机器人学、物理学、人工智能等学科领域的理论、方法和技术,生成尽可能逼真的角色形状和运动。人体动画的研究内容包括角色造型、角色运动形变、运动控制、表情动画、毛发动画、服装动画、自然语言驱动的动画、运动捕获、运动重现和自主运动等。

人体是由关节构成的,关节动画是人体运动控制的核心。驱动关节链结构运动的方式主要有两种:运动学方法和动力学方法。运动学方法仅考虑物体的位置、速度和加速度等运动参数,这种运动与物体的质量、产生该运动的隐含力无关。而在动力学方法中,运动参数完全由动力学方程所决定。关节结构由一系列刚体链在关节处连接而成,其中链结构的自由端称为末端影响器。虽然在机器人学中有平移关节和旋转关节两种类型,但在计算机动画中只有旋转关节。在数学上,描述关节链的常用方式为 DH 表示,它的特点是将坐标系附在每一个链上来描述一个链相对于其他相邻链的运动,两个相邻链坐标系的齐次坐标变换矩阵称为 A 矩阵。在正运动学中,所有关节向量的值由动画师显式给出,末端影响器的位置由关节间变换的复合得到。而逆运动学是一种目标导向的运动指定方法,简称 IK。在逆运动学方法中,动画师指定末端影响器的位置,系统求解关节向量的值。在动力学方法中,物体的运动是由力和力矩驱动的,由于计算所得的运动是符合物理规律的,故能生成更复杂和逼真的运动。动力学方法又可分为正动力学方法和逆动力学方法。在正动力学方法中,给定作用于物体的力和力矩,计算各关节的位移、速度和加速度;而在逆动力学方法中,力和力矩是通过给定的运动学参数反求得到的。

目前一种非常流行和实用的人体动画技术是运动捕获和运动重现。与通过交互方式设置关节动画的方式不同,运动捕获采用软、硬件系统自动记录表演者的真实运动信息,并把动作结果施加到一个虚拟的人或动物上。该方法类似于早期 Disney 公司制作卡通片《白雪公主》时使用的 rotoscoping 技术,即动画师根据素材画面,用手工交互的方式跟踪获取画面主体的运动信息。运动捕获可以获取表演者动作的个性和细节,是生成逼真人体动画最实用、最

有效的方法。该技术不仅可以提高角色动画的质量,而且极大地缩短动画制作的周期。捕获的人体关节运动数据通常采用 BioVision 公司的 BVH 格式来储存。运动捕获主要包括光学运动捕获系统、磁性运动捕获系统和机械式运动捕获系统。运动捕获的数据是表演者真实运动的映像,但表演者和虚拟人的关节结构数据(如身高、腿长、手臂长度等)不一定完全相同。若不加修改地直接把运动捕获的数据直接应用到一个不同尺寸的虚拟人上,就有可能出现运动不协调、双脚离地等不真实的运动。运动重现可以把一个角色的动画赋给另一个具有相同关节结构但具有不同关节长度的角色,并能保持原有动画的质量。

人体动画是现代高科技电影的核心技术之一,它使得用计算机生成真实影视产品成为可能。导演可使用计算机指挥、调度各个人造角色、舞台场景、灯光和摄像机,在一个虚拟世界里产生出真实的影视佳作。人体动画在影视业、娱乐业、广告业、体育训练、舞蹈教学、医学等多种领域得到了广泛的应用。

参考文献

1. 鲍虎军,金小刚,彭群生等. 计算机动画的算法基础. 杭州:浙江大学出版社,2000
2. Watt A, Watt M. Advanced Animation and Rendering Techniques: Theory and Practice. London: Addison-Wesley, 1992 (金小刚)

ren zhu jiyi

人助机译 (human-aided machine translation, HAMT) 机器翻译的一种类型。翻译工作的主要部分由计算机程序自动完成,但需要人做一些辅助的乃至关键性的工作以帮助计算机克服知识不足的缺陷,以提高译文的质量。

全自动高质量的**机器翻译**(MT)诚然是最理想的,但由于自然语言固有的复杂性,当代计算机体系结构的制约以及机器翻译的理论与技术尚不够成熟,全自动高质量的机器翻译系统是不存在的。实际运转的自动翻译系统基本上是人助机译的。人助机译又可分为批处理与人机交互两种方式。批处理方式指人只做译前编辑或(和)译后编辑,翻译程序一旦启动就自动运行。人机交互方式指程序运行中遇到解决不了的问题时,就请求人给予帮助。这些问题包括文本中出现的未定义词(词典中未登录的词或虽登录了但缺少语法语义信息)或仅依靠词典

与规则库无法消解的歧义结构。人助机译系统可以引入**知识获取与机器学习**的功能,以不断提高自动化的水平,扩大可以翻译的范围和提高译文的质量。

参考文献

- 冯志伟,杨平. 自动翻译. 上海:知识出版社, 1987 (俞士汶)

renwu diaodu chengxu

任务调度程序 (task scheduler) 并发程序设计系统中用于调度和分派处理器的程序。又称进程调度、处理器调度或低级调度程序。

在并发程序设计系统中,在同一时刻可能有多个**进程**同时竞争处理器,这就需要按任务调度算法,把处理器占用权交给就绪队列中的一个进程,以便让它执行。任务调度的主要功能有:记录进程状态和维护进程状态队列、分派处理器和回收处理器。

在单处理器系统中,任务调度算法主要解决如何按一定的策略把空闲处理器分派给一个就绪进程。评价一个任务调度算法的主要依据有:系统吞吐率和响应时间。常用的任务调度算法有:

(1) **优先数调度** 优先数调度算法是基于设置的进程优先数,把处理器分派给就绪队列中优先数最高的就绪进程。

设置进程优先数的方法可分为:静态和动态优先数法。静态设置方法是指在创建一个进程时,按照该进程的进程类型、资源需求等因素由系统指定一个优先数,而且,在进程整个生命周期该优先数保持不变。这种方法易于实现、开销较小,但不够灵活、效率较低。常用于**批处理**操作系统。动态设置方法是指进程生命期内,由系统根据进程占用处理器的时间长短、进程任务的缓急程度、资源的均衡使用等因素动态地改变进程优先数。这样能获得更精确的调度和更佳的调度性能。

(2) **轮转法调度** 时间片轮转调度算法是轮流调度所有就绪进程,即每隔一个时间片,依次从处于就绪状态的诸进程中选一个运行。该算法主要用于分时系统。

轮转法调度进程的关键一是要利用闹钟,定时发出时钟中断,以调度另一就绪进程运行。二是决定时间片大小,时间片过大,退化为优先数法,难以实现轮转执行。时间片过小,导致调度和分派进程过于频繁,增加系统开销。时间片大小的确定应综合考虑多种因素,如:系统响应时间、联机终端个数、

处理器处理速度及系统的其他处理能力。

根据时间片的大小、就绪队列的个数,可把轮转法调度分成简单循环轮转调度、变长时间片轮转调度、多级反馈队列轮转调度。

(3) 分级调度 又称反馈队列或多级队列调度。该调度算法的主要思想是将就绪进程列入多个不同级别的就绪进程队列。任务调度每次从高级就绪进程队列中选取可占用处理器的进程,只有在选不到时,才从较低级就绪进程队列中选取。进程分级可事先规定,例如,使用外围设备频繁者属于高级;分时系统中的终端用户进程定为高级。进程分级可动态进行,例如,凡运行超越时间片后,就进入低级就绪队列,以后赋给较长的时间片;凡启动磁盘、磁带而成为等待的进程,在等待结束后就进入中级就绪队列等。

随着分布式系统的发展,分布式任务调度算法成为研究的热点。其研究内容之一是设计将一组相互协作的任务分配到一组处理器上运行的分配算法。分布式任务调度算法的评价标准与单处理器系统相比,增加了对负载的要求,即要求系统中各处理器具有较为平衡的负载,避免出现个别处理器特别繁忙,而其他处理器非常空闲的情形。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989 (郑宇华)

rongcuo jisuan

容错计算 (fault-tolerant computing) 计算机在发生内部故障情况下,仍能正常运行并给出正确结果的计算能力。又称“容错技术”。

容错是指容许系统在运行过程中发生一定的差错或故障时仍能保持正常工作而不影响正确结果的一种性能或措施。具有容错功能的系统称为容错系统,该系统在出现某些硬件故障或软件差错时能自行检测和排除故障或差错,以保证完成预期的任务,并得到正确结果,整个过程不需要用户的干预,因而对用户来说是透明的。

容错与可靠性有着密切的关系,可靠性是系统性能评价中的主要指标之一。容错计算则是提高系统可靠性的有效手段和措施,通过容错措施的实施可以大大地提高可靠性。但是容错不是提高可靠性的惟一措施。例如一种与容错的概念恰恰相反的措施称为避错技术也是提高可靠性的一种有效措施。

容错计算的思想产生较早。在早期的计算机系统中已广泛地使用纠错编码、故障检测等技术。20世纪50年代初期,在贝尔继电器计算机上就使用过双处理器。其中一个处理器作为正常运行的部件,另一个则作为备份处理器。一旦正常运行的处理器发生故障,则从下一条指令开始就由备份处理器执行。从而提高了系统的可靠性。实际上就是冗余技术的萌芽。此后,不少容错措施,如复执功能、双工、双机系统、三模冗余系统等也开始研制。但是容错计算的全面发展是在容错理论建立并发展的基础上展开的。

从60年代到70年代,容错计算理论和技术有了飞速的发展。出版了有关容错理论的期刊、杂志,举办了国际性的学术年会(如国际容错计算会议FTCS),出现了容错计算机制造商(如美国的TANDEM公司和STRATUS公司等),并研制了所谓“不停机计算机”、“零故障计算机”等容错计算系统。由于容错计算系统的成本较高、要求较严,因此在中国当前容错计算系统尚未十分广泛。

一个完整的容错计算系统至少包括下列10个组成部分:故障限制、故障检测、故障屏蔽、复执、故障诊断、重配置、恢复、重启、修复和重构等。简要分述如下:

(1) 故障限制 一旦系统发生故障,要求限制该故障的影响范围。将由于故障引起的“污染”限定在系统的最小范围内而不使其扩散。故障限制可以用故障检测、一致性检查等方式完成。它既可以由硬件,也可以由软件来实现。

(2) 故障检测 故障检测技术一般分为两大类,脱机检测和联机检测。在进行脱机检测时,被测系统不能正常运行。例如在系统上运行诊断程序时必须暂停执行系统的应用程序。而联机检测则提供实时检测的功能,此时检测与正常应用程序可同时运行。

(3) 故障屏蔽 是将已发生的故障隐蔽起来,不使暴露。对用户来说觉察不到故障的发生。故障屏蔽与故障检测正相反,前者是掩盖由某些故障引起的错误信息,后者则要激活故障使其产生错误信息,以便检测到故障的存在。故障屏蔽多采用冗余技术来实现。如多数裁决器。

(4) 复执 在计算机硬件系统中的故障可以分为两类:永久型和暂时型。永久型故障一旦发生不会自行消失。而暂时型故障又分为两种,间歇型和瞬时型。前者的发生为断续且不定周期的,而后者

为偶发一次性的。复执技术正是针对瞬时型故障的偶发性特点而设计的。当因瞬时型故障而产生错误时,重复多次执行相同的指令可能会因故障消失而得到正确的结果。因此,复执是一种代价较小效果较好的容错计算措施。

(5) 故障诊断 故障诊断包括两部分:故障检测和故障定位。故障检测已如上述,故障定位则是对故障检测的进一步加强,即不仅要检测出故障的存在而且要确定故障发生的部位。在容错计算系统中,必须能确定故障部位以进行纠正或修改,从而获得最终的正确结果。

(6) 重配置 当故障被检测并定位以后,就需要对故障部件作出相应的处理,这称为系统的重配置。重配置方法有两种:一是用备份部件代替故障部件(如动态冗余技术等);二是直接将故障部件从系统中分离出来,此时系统仍能正常工作,但功能已有所减弱,称为降级使用或称为正常降级。

(7) 恢复 系统发生故障并经重配置后,将运算回复到发生故障前的某一断点。这种恢复的方法称为回退法。它是利用备份文件、校验点以及流水日记账等方法记录其运行轨迹。然后回退到适当的断点,重新恢复所需的数据等。这里要注意的是必须确定故障的潜伏期。若发生的故障具有一段潜伏期,则在回退时必须回退到该故障的潜伏期前,以免再次遇到未被检测出来的故障而造成的错误。

(8) 重启动 由于发生了故障,系统中的信息可能多处遭破坏甚至无法恢复原来的信息。因此需要进行重启动。重启动一般分为三类,热启动、温启动和冷启动。热启动是指在断点前的所有信息均未遭破坏直接从断点开始的启动。温启动是指在断点前有某些信息被破坏,但这些已破坏的信息可以通过备份或记录来恢复,恢复后可以自断点进行启动。冷启动则是指所有信息都必须重新装入后才能进行的启动。

(9) 修复 确定故障部件后必须进行修复。修复方式有脱机修复和联机修复两种。脱机修复是指故障部件与系统分离情况下的修理;联机修复是指系统自动切除故障部件换上备份部件,继续工作。或自动切换故障部件,使系统降级继续运行。

(10) 重构 当所有故障部件被替换并重配置以后,系统应回复到初始的基本结构状态。例如三模冗余电路,在发现有一个模块发生故障后,应及时切换下来,并将备份模块投入运行,重新构成三模冗余电路,以保证系统的可靠性。

容错技术可以分为3类,即故障检测技术、屏蔽冗余技术和动态冗余技术。

故障检测虽不直接提供容错功能,但它可在发生故障时发出警告。由于实现这种技术的成本较低且较容易,因此常被应用于微、小型计算机设计中。这些系统往往将故障检测技术与联机检测相结合,检测虽是检错,而不是容错,但它是容错计算的前提,没有检错功能,也就无法进行容错计算。

屏蔽冗余又称静态冗余,它与故障检测功能相反,对故障或错误的出现不作警告处理,容许错误存在而保持系统的正常运行。例如纠错编码和多数裁决冗余电路等,都是屏蔽冗余技术的具体实施方案。这些技术在容错计算系统中使用十分广泛。

动态冗余是在静态冗余电路的基础上根据系统在运行过程中发生的故障随时改变系统结构的冗余技术。动态冗余技术实际上是静态冗余、联机故障检测及重配置等几项技术的结合。它在静态冗余电路工作的同时,对系统作检测并测得的故障模块自动切换或将系统降级使用,因此该技术具有很强的实时性和容错功能,是高可靠系统容错设计的重要技术之一。但是实现这种技术的成本较高。对系统的运行速度也有一定的影响。

近年来流行的高可用系统是一种成本较低的容错系统,主要用于避免软件错误造成的停机,两台或多台机器互为备用机,平时各执行不同的任务。但一台机器出错时应用程序迅速切换到另一台备用机上运行。

容错系统是20世纪70年代初开始在国际上发展起来的,至今仅有约二十年的历史。随着微电子技术及计算技术的不断的迅猛发展,对容错计算的理论和计算技术将会提出更高的要求同时也会得到更广泛的应用。用容错计算的策略提高可靠性是一个十分积极而又实际的思想。容错计算的研究将会有很大的发展。

参考文献

1. Daniel P Siewiorek, Robert S Swarz. The Theory and Practice of Reliable System Design. Digital Press, 1974
2. Barry W Johnson. Design and Analysis of Fault-Tolerant Digital Systems. Addison -Wesley Publishing Company Inc., 1989

(徐拾义)

rongcuo jisuanji

容错计算机 (fault-tolerant computer) 在

硬件发生故障或软件产生错误时仍能继续运行并完成其既定任务的计算机系统。

容错计算机的主要设计目标是为了提高计算机系统的可靠性、可用性和可信性等性能。提高计算机可靠性的方法可以分为两大类:一类是排错技术,主要是通过使用可靠性高的元器件,严格的老化筛选等方法达到尽量减少发生故障的可能性;另一类是容错技术,主要是运用冗余技术来抵消由于故障而引起的影响。所谓冗余技术,简单地讲,是在正常运行所需的基础上加上一定数量的信息、时间或后备硬件、后备软件的方法。冗余技术是容错计算机中容错技术的基础。冗余大致上可以分为下列几种类型:

(1) 硬件冗余 以检测或屏蔽故障为目的而添加一定硬件设备的方法;

(2) 软件冗余 为了检测或屏蔽软件中的错误而添加一些在正常运行时不需要的软件的方法;

(3) 信息冗余 在实现正常功能所需的信息以外,再附加一些信息的方法,例如纠错码就是信息冗余的一种形式;

(4) 时间冗余 使用附加一定的时间来完成系统的功能,这些附加的时间主要是用在故障检测或故障屏蔽上。

最常用的硬件冗余是硬件的重复。硬件冗余一般可以分为3种类型:静态冗余(也称为被动冗余)、动态冗余(也称为主动冗余)和混合冗余。静态冗余将已发生的故障屏蔽起来,使不影响运行的结果。被动冗余主要是依靠表决机制来屏蔽发生的故障,因而这种方法不需要故障检测也不必进行系统的重新配置等就可以获得容错的效果。被动冗余技术中使用最广的是三模冗余 TMR。TMR 的基本概念是使用3套完全相同的硬件系统执行相同的任务,然后由1个多数表决器对这3套系统的输出进行表决以确定整个系统的输出。多数表决器的表决原则是三中取二。也就是说三模冗余系统可以容许有1个模块发生故障而不至于影响到整个系统运行的正确性。三模冗余的关键是多数表决器本身的可靠性问题。提高多数表决器可靠性的方法有多种,其中最常用的方法是多数表决器本身也使用三模冗余,即利用3个独立的多数表决器,每个多数表决器分别接受来自3个模块的输出作为它的输入,然后再分别输出。这种系统通常被称为带三重多数表决器的三模冗余系统。除了三模冗余系统外,还有多于三模的冗余,称为 N 模冗余。主动冗余技术与被

动冗余技术相反,它是通过故障检测、故障定位及故障恢复等手段达到容错的目的。因而在主动冗余技术中不是去防止故障引发的错误,而是暴露由故障引发的错误,从而去纠正错误。主动冗余技术中最典型的方法是构造带有比较器的双工系统。在这种方法中,使用两套完全相同的硬件,且同时完成完全相同的任务,然后对它们的结果作比较。当然,仅仅有1个比较器的双工系统只能检测到有无故障,尚不足以确定哪一个模块出了故障。所以在这样的系统中还必须增加一定的措施才能作故障定位。动态冗余技术除了上述方法以外,还有诸如热备份、使用把关定时器等都是较为常用的方法。硬件冗余的第三种类型是混合冗余。这种技术是将主动冗余和被动冗余结合起来,且取二者之长处。它先使用被动冗余中的故障屏蔽技术,使系统免受某些可以被屏蔽的故障的影响。而对那些无法屏蔽的故障则采用主动冗余中的故障检测、故障定位、故障恢复等技术,并且对系统可以作重新配置。因此,混合冗余的效果要大大优于主动和被动冗余。然而,由于混合冗余既要有被动冗余的屏蔽功能,又要有主动冗余的各种检测、定位等功能,它的附加硬件的开销是相当大的,所以混合冗余的成本很高,仅在对可靠性要求极高的场合中采用。混合冗余的方法也有多种,例如,带热备份的 N 模冗余技术,自清洗冗余技术、筛选模块冗余技术等。

信息冗余是一种将冗余信息添加到数据上从而达到故障检测、故障屏蔽和容错的目的。信息冗余最好的例子就是检错码和纠错码。这是将冗余的信息加到一个数据字上使每一个数据字变为一个新的带有冗余信息的字。这种冗余信息的添加方法是按照一组预定的规则进行的。符合添加规则而形成的带有冗余信息的字称为码字,而那些虽带有冗余信息但不符合添加规则的字则称为非码字。按添加冗余信息的规则加上冗余信息的过程称为编码。反之,将已编码的字恢复成原来形式的过程则称为译码。一般来说,经过编码的码字只是全部编码的子集,另一部分则是非码字,当系统出现故障时,可能会将码字变成非码字,于是在译码过程中会将引起非码字的故障检测出来。这就是检错码的基本思想。至于纠错码则不仅可以将错误检测出来,而且还能将由故障引起的非码字纠正成正确的码字。由此可见,信息冗余的主要任务在于研究出一套理想的编码和译码技术来提高信息冗余的效率。编码技术中最简单、最常用的检错码是奇偶校验码。奇偶

校验的基本思想是在二进制的信息字上附加一位冗余位,称为校验位,使得该码字(这里的码字是信息位加上冗余位而形成的信息字)中所含有的1的个数为偶数或为奇数。如果码字中的1的个数为偶数,则称这种校验为偶校验。如果码字中的1的个数为奇数,则称这种校验为奇校验。由于奇偶校验码简单实用,便于硬件的实现,因而在计算机系统中被广泛使用。但是奇偶校验码存在一定的缺点,例如它不能检测偶数个同时发生的故障,因而在它的基础上又发展了多种不同的奇偶校验码,例如分段奇偶校验码、分字节奇偶校验码等。除此以外,还有“n中取m”码、双重码、检查和以及循环码等都是常用的检错码。汉明码(参见存储器差错校验)是纠错码中最典型的代表。它不仅能够检测出1个甚至两个故障,而且还能将错误纠正过来,将非码字改正为码字,因而被广泛采用。

时间冗余是以时间(即降低系统运行速度)为代价以减少硬件冗余和信息冗余的开销来达到提高可靠性的目的。在某些实际应用中,硬件冗余和信息冗余的成本、体积、功耗、重量等开销可能过高,而时间并不是太重要的因素时,可以使用时间冗余。时间冗余的基本概念是重复多次进行相同的计算,或称为重复执行,简称复执,以达到故障检测的目的。实现时间冗余的方法很多,但是其基本思想不外乎是对相同的计算任务重复执行多次,然后将每次的运行结果存放起来再进行比较。若每次的结果相同则认为无故障;若存在不同的结果则说明检测到了故障。不过,这种方法往往只能检测到瞬时型故障而不宜检测永久型的故障。这是因为瞬时型故障会使各次运行产生不同的结果。若不仅要检测瞬时型故障,而且还要检测固定故障等永久型故障,则单靠时间冗余是有困难的。因此,在系统中还必须附加少量的冗余硬件。时间冗余与硬件冗余的结合,既能检测瞬时型故障,又能检测永久型故障。

软件冗余是利用冗余的软件来检测硬件和软件故障的方法。利用冗余软件进行故障检测的方法很多。常用的有一致性检查、能力检查和多版本程序设计等。一致性检查是对某一运行结果先作一定的预测,然后在程序运行中和运行后对其结果与预测的结果作比较。若实际结果在期望值的范围内,则认为正常,若实际结果超越了期望值的范围,则认为有故障。能力检查是用检查程序去检查系统中各个部件应有的能力,例如用程序来读写某一个存储单

元,以检查该单元的存储和读写能力,又如用一组特定的数据去检查运算逻辑部件,以判断该部件能否进行正常的运算等。多版本程序设计是对一个相同的任务(或算法)用不同的方法进行程序设计,然后对不同版本的程序运行后得到的结果进行比较,若所有版本运行的结果相同,则认为无故障,否则,就认为有故障存在。值得注意的是,这种方法实际上是来自于硬件冗余技术中的N模冗余的思想。多版本程序设计不仅能检查硬件故障,也可以检查软件本身的故障,因此,在软件容错技术中经常使用。

上述的冗余技术,即硬件冗余、信息冗余、时间冗余和软件冗余是使系统获得容错功能和提高可靠性的基本措施和手段。在实际应用中,上述4种冗余技术经常是结合起来使用的。将这些冗余技术融合在一个计算机系统中,就称这个系统为冗余系统。

一般说来,一个较为完整的冗余系统,在处理运行中出现的故障时,大体上有以下10个步骤:

(1) 故障检测 这是处理故障的基础,因为要容错就先要将故障检测出来。故障检测的方法很多,如上述的奇偶校验就是检测故障的一种方法。故障检测一般分为两类:联机检测和脱机检测。前者提供了实时检测的能力,这种检测工作与系统的正常工作同时进行。后者在进行检测时,系统必须停止正常工作。

(2) 故障屏蔽 这与故障检测正好相反,它不是将故障检测出来,而是将出现的故障屏蔽起来,使系统不受故障的影响。

(3) 故障限制 限制故障影响的范围,防止已发生的故障影响到系统的其他部分。

(4) 复执 这是一种检测瞬时型故障的有效措施。它可以提高计算机抗瞬时型故障干扰的能力。

(5) 故障诊断 在故障检测的基础上,对故障进行定位。这对以后的修复、重配置等有很重要的意义。

(6) 系统重配置 若故障一旦被检出并定位,系统应有能力将发生故障的元件或部件替换下来,或将故障部件与其他部分隔离开来。当故障部件被替换下来后,系统中可能缺少了这一部件,但系统仍能保持正常运行,只是系统运行速度下降、功能减弱。这一现象称为系统降级使用。

(7) 系统恢复 当检测出故障,必要时在系统重配置后即可消除故障引发的差错。这时,系统应能返回到出现故障断点前的情况继续运行。这个过程称为系统恢复。

(8) 系统重新启动 如果系统由于出现过多的故障而造成大量的错误,以致破坏了许多无法恢复的信息时,就不能再使用上述的系统恢复的办法,而必须重新启动运行。重新启动分为热启动和冷启动。前者是在部分信息遭到破坏但还有一部分可以利用的情况时使用,而后者则是在几乎所有信息均遭破坏的情况下使用。

(9) 修复 凡是已确定有故障的部件必须进行修复。修复分为脱机修复和联机修复二种。若要修复的部件卸下后对系统影响不大,或者修复这些部件时系统必定会停机,就使用脱机修复。联机修复通常是指系统能自动启用备份部件替代有故障部件,并保持系统继续运行,然后再修复切换下来的故障部件。

(10) 系统重组 当上述各步完成后,系统必须重新组合,以便完全恢复正常运行。

容错计算机主要应用于工业生产、医疗、航空、航天、军事、公安、交通、金融、机要等部门对计算机的可靠性要求很高的场合。在应用需求的推动下,容错计算机的理论和技术在不断发展。尤其是在硬件和软件容错理论、测试算法、诊断技术等方面,尚需继续深入研究。

参考文献

1. Barry W Johnson. Design and Analysis of Fault-Tolerant Digital Systems. Addison Wesley Publishing Company Inc., 1989
2. Daniel D Siewiorek, Robert S Swarz. The Theory and Practice of Reliable System Design. Digital Equipment Corporation, 1982 (徐拾义)

rouxing zhizao xitong

柔性制造系统 (flexible manufacturing system, FMS) 通过开发多用途、可组合的加工模块,并使用计算机技术将物流和信息流有机地结合起来,能根据制造任务或生产环境的变化自主迅速调整的高效自动化制造系统。柔性制造系统的柔性体现为:机床柔性(数控)、产品转化柔性、加工方式柔性、工序互换柔性、连续运行柔性、产量柔性、扩展柔性和生产柔性等;它具有自动加工及自动换刀,自动检测、自动控制质量和故障诊断,自动变换加工程序,自动完成多种零件或零件组加工,对加工顺序及生产节拍的自适应调整等特点,可高效率地制造多品种小批量的产品,并能同一时间用于不同的生产任务,从而实现加工系统从“刚性化”向高度“柔

性化”的转变。因此,柔性制造系统能够以制造大批量产品的速度和成本来生产各种不同型号的小批量产品。

20 世纪 50 年代自动化流水生产技术和数控机床发展很快,流水生产技术解决了大批量生产问题,数控机床的出现则为单件小批量生产提供了技术保证。在此基础上,英国 Molins 公司的雇员 Theo William-son 提出了解决多品种变批量生产难题的方案,就是采用柔性制造系统。世界第一套柔性制造系统(FMS)——M24 系统于 1967 年在英国诞生。随着数控机床和加工工艺技术的发展、计算机管理和控制水平的提高、自动化配套设备和各种传感器及检测元件的大量应用,FMS 在 20 世纪 70 年代发展很快。80 年代末,FMS 已进入了实用化阶段。

柔性制造系统(FMS)一般由以下几部分组成:

(1) 加工系统 一般包括对原材料或毛坯进行加工的通用自动机床,如数控机床、加工中心,以及数控组合机床。除了在机床上有自动换刀系统(包括换主轴箱)外,还应在其附近设置自动更换夹具和工件的系统。

(2) 物流存储系统 主要包括存储夹具和毛坯的自动化仓库,通用输送系统(如无人小车、输送带、工业机器人等),装卸工件和夹具的装卸系统(如机械手、工业机器人和自动化装卸小车)等。

(3) 运行控制系统 主要完成工件加工程序或数控程序的自动存储、检索、处理和分配到各有关机床,并完成对 FMS 的生产、运输、换刀和自动存储库的控制。通常利用计算机作为控制系统的中央控制机。

(4) 质量保证系统 主要完成 FMS 的工作情况监测,运输系统的监视,加工质量的自动检测等工作,以保证系统正常运行及加工的质量。

(5) 信息处理系统 它通过计算机及通信网络、数据库等,实现信息自动的收集、传递、交换与处理。应用 FMS 后,能有效地缩短加工系统的调整时间,减少切削和辅助时间,显著节约劳动力,有效提高生产率和设备的利用率,降低加工费用,增加经济收益。FMS 可以在设备和技术规范范围内自动适应加工工件和生产批量的变化,具有较高的生产率和对产品及市场的应变能力,适合于制造多品种、中小批量零件,在制造领域将得到越来越广泛的应用。

根据柔性制造模块的数目和它们的配置,可把柔性制造系统分成以下几个等级:①柔性制造模

块;②柔性制造单元;③柔性制造系统;④柔性制造工厂。企业实施柔性制造系统需要大量投资。柔性制造系统在一些离散型制造企业得到应用,大幅提高了生产效率。但由于其高科技、高投资、对外部供销企业要求高等特点,使其在出现后的30多年里并没有获得很大发展,甚至应用成功的企业少于应用失败的企业。一般来说,企业采用柔性制造系统需要具备以下条件:①供销环境要求采用柔性生产,并且市场潜力足以使企业满负荷运转;②拥有一支掌握柔性制造技术的员工队伍,能够保证制造体系的正常运转;③要有资金上的充分保证。

参考文献

1. 谭益智等. 柔性制造系统. 北京: 兵器工业出版社, 1995
2. 徐兰如等译. 柔性制造系统手册. 北京: 宇航出版社, 1987
3. 吴季良. 柔性制造系统实例. 北京: 机械工

业出版社, 1989

(潘晓弘 顾新建 郭意杰)

ruancipan ceshi

软磁盘测试 (floppy disk testing) 为保证软磁盘数据可靠存储和交换,对软磁盘的质量和性能所进行的检测。软磁盘测试主要包括对软磁盘机械尺寸、物理性能和电磁性能的测试。

测试标准 国际标准化组织(ISO)先后制定了一系列对应于不同规格软磁盘的国际标准,详细规定了软磁盘的机械尺寸、物理性能和电磁性能及其测量参数的范围,作为软磁盘测试的基本依据。我国的软磁盘国家标准等同采用了ISO软磁盘国际标准,并已陆续颁布了一些软磁盘国家标准。表1列出了130 mm(5.25 in)软磁盘和90 mm(3.5 in)软磁盘标准主要的测试项目。

表1 软磁盘标准主要测试项目

规格	盘径/mm	130	130	90	90
	未格式化容量	500 kB	1.6 MB	1 MB	2 MB
	ISO 国际标准号	ISO 7487	ISO 8630	ISO 8860	ISO /IEC 9529
	国家标准号	GB 9416	待颁布	GB /T 13719	待颁布
机械尺寸测试		磁盘结构尺寸 封罩结构尺寸	磁盘结构尺寸 封罩结构尺寸	磁盘和盘毂结构尺寸 外壳和快门结构尺寸	磁盘和盘毂结构尺寸 外壳和快门结构尺寸
物理性能测试		可燃性 启动转矩 运行转矩 磁盘的线性热膨胀系数 磁盘的线性湿膨胀系数 磁盘和封罩光透射率	可燃性 启动转矩 运行转矩 磁盘的线性热膨胀系数 磁盘的线性湿膨胀系数 磁盘和封罩光透射率	可燃性 启动转矩 运行转矩 磁盘的线性热膨胀系数 磁盘的线性湿膨胀系数 外壳允写窗口光透射率	可燃性 启动转矩 运行转矩 磁盘的线性湿膨胀系数 磁盘的线性湿膨胀系数 外壳允写窗口光透射率
电磁性能测试		典型磁场强度 平均信号幅度 分辨率 重写 调制 漏码 冒码	典型磁场强度 平均信号幅度 分辨率 重写 调制 漏码 冒码	典型磁场强度 平均信号幅度 分辨率 重写 峰值位移 调制 漏码 冒码	典型磁场强度 平均信号幅度 分辨率 重写 峰值位移 调制 漏码 冒码

软磁盘的机械尺寸测试的主要目的是保证软磁盘外形尺寸的一致以及在软磁盘驱动器上使用的可互换性,物理特性测试用以规范软磁盘材料的性能。除表1列出的有关项目外,磁盘的表面电阻和磨损寿命也是软磁盘物理特性的两项较重要的测试项

目,目前国际标准还未作具体规定。电磁性能测试用来校验软磁盘记录信号的质量,其中典型磁场强度、平均信号幅度、分辨率、重写、调制测试项目校验磁盘表面质量状况,漏码、冒码测试项目主要校验磁盘记录磁道的缺陷状况。

测试设备 软磁盘机械尺寸测试通常使用专用的量规,如厚度规、落通规、孔径规等进行测量。这些量规在使用时,一般可以由量规上千分表直接读出尺寸误差值,但有的需要由被测软磁盘通过量规的通端和止端的情况反映尺寸公差。

软磁盘物理特性测试涉及不同的物理量,需要使用不同的测量仪器。软磁盘转矩和光透射率需要使用专用的软磁盘转矩仪和软磁盘透光性测定仪测试。软磁盘的线性湿热膨胀系数则要使用热机械分析仪和差示扫描量热计等通用仪器测量。在测试方法上,这类仪器基本上都是将被测的物理量通过转换装置变为易于测量处理的电信号,再经过放大处理,由仪表指示或进行数字化后直接显示数字量值。

用于软磁盘电磁性能测试的主要有软磁盘分析仪和软磁盘鉴定仪两种设备。软磁盘分析仪用于对少量的软磁盘样品的质量及生产工艺作分析、评价。它除可以测试国际标准中有关电磁性能的全部项目外,还具有较强的辅助分析功能。它的测试精度比较高,可以提供详细的测试数据报告和直观的图形曲线,但测试速度比较慢。软磁盘鉴定仪用于软磁盘生产线上对产品筛选分档。它一般只检测对软磁盘实际使用有严重影响的磁道质量参数,如漏码、冒码。它的特点是测试速度快,操作简便,一台仪器可以同时检测几片到十几片软磁盘,测试结果只判定和指示出被测磁盘所属的质量等级。这两种仪器的测试原理基本是相同的。它们都是在被测软磁盘上写入特定的记录信号(全1或全0),然后读出这些信号,通过评价每位信号的幅度和它们的平均值及其相互关系得到反映软磁盘电磁性能的一系列测试参数。

参考文献

刘文伯,周希瑾. 微型计算机用软磁盘. 北京: 科学出版社, 1991

(李健)

ruancipan cunchuqi

软磁盘存储器(floppy disk storage) 利用软磁盘作存储媒体,磁头作电磁转换器件,进行数据记录的存储设备。它是旋转磁表面型存储设备的一种。软磁盘是以圆形聚脂膜为基底,在它的上面涂敷磁性层材料的数字记录媒体。

软磁盘存储器由**软磁盘驱动器**、**软磁盘控制器**、**软磁盘适配器**、软磁盘和相关软件(驱动程序和管理程序)等部分组成。软磁盘驱动器的功能是驱动软盘片按一定转速稳定旋转,驱动载有磁头的存取

臂到达且稳定在指定的半径位置上,控制磁头在盘面磁层上按一定的记录格式和编码方式进行写入和读出。软磁盘控制器是控制软磁盘驱动器实施数据的存入与取出的设备。它按照主机发来的命令控制驱动器进行磁道的寻找(存取臂定位)、头的选择、数据的写入(存)与读出(取)等操作。软磁盘适配器则用来进行接口匹配,使同一种规格的软磁盘可与各种规格的系统总线相连接。

1972年IBM 3470型数据输入系统问世,在此系统中使用了记录媒体可换的外径为8英寸、单面、单密度的软磁盘和相关设备。4年后,软磁盘的外径减小到5.25英寸。软磁盘存储器不仅可用作辅助存储器,而且可用于数据输入,特别适用于个人计算机和办公自动化系统,因而发展十分迅速。成为微型计算机和其他电子产品不可缺少的重要设备。1980年后,又陆续出现了3英寸~4英寸的几种软磁盘,经美国国家标准学会按3.5英寸规格标准化后,这一小型软磁盘规格一直沿用至今。

软磁盘有多种分类方法。按外形尺寸分,有8英寸、5.25英寸、3.5英寸和2英寸等。按记录密度分,有单密度(采用调频制记录)和倍密度(采用改进调频制记录)两种。按记录材料分,有涂敷型普通铁氧体、涂敷型钡铁体、涂敷型金属磁粉和溅射型金属连续薄膜等。按记录方式分,有纵向(水平)记录和垂直记录两种。依据盘片尺寸、磁道密度、位密度、记录材料以及记录编码的不同,不同类型的软磁盘存储器有不同的存储容量。3.5英寸、双面、倍密度、纵向记录、容量为1.44 MB的软磁盘存储器是常用的。

软磁盘存储器由于结构简单、价格低廉、便于脱机存储和携带,尤其是在互换性的标准化上做得好,成为目前最常用的交换媒体和输入媒体。但单片软磁盘存储器的存储容量太小,使它的使用受到限制。其后虽然有各种大容量的软磁盘规格提出,如容量为8 MB的伯努利软磁盘,容量为21 MB的光软磁盘,120 MB的LS-120软盘,240 MB的Zip软盘等,但由于种种原因,特别是标准化程度差,没能得到普遍的使用。近来,作为交换媒体,在许多应用领域,软磁盘已被可改写光盘、半导体盘和可换硬盘等所取代。

(林兼)

ruancipan kongzhiqi

软磁盘控制器(floppy disk controller) 连接计算机主机与软磁盘驱动器,以实现主机对软磁

盘驱动器进行数据存取的控制部件。它将主机需要存储的数据,经并串转换、调制码和检纠错码编码等处理后,送往软磁盘驱动器中由计算机程序指定的地址处进行记录;或从软磁盘驱动器取出主机需用的数据,经调制码和检纠错码译码及串并转换等处理后,送往计算机主存;并及时检测软磁盘控制器中的状态信息,提供给主机处理。

软磁盘控制器由主机接口、控制逻辑和驱动器接口三部分电路组成。

(1) 主机接口电路一般设置若干个寄存器及直接存储器存取(DMA)、中断等逻辑电路,它为软磁盘驱动程序提供接口,驱动程序通过相应端口对寄存器寻址,实现命令、数据和状态的传送。

(2) 控制逻辑实现数据的转换、处理和寻道等控制信号的产生,它由两条信息通路组成。一条是数据信息通路,用于处理读、写数据。该通路包括数据驱动和缓冲、串并和并串转换、调制码和检纠错码的编译码、写预补偿和数据分离电路等环节。另一条是控制通路,它配合读、写过程,控制数据在数据通路各环节的流动,并根据不同的命令和状态信息形成驱动器接口所需的控制信息序列。

(3) 驱动器接口逻辑由发送和接收驱动电路组成,在控制器和软磁盘驱动器间传送控制信息、状态和数据。控制信息主要有设备选择、主轴电机启动、磁头选择(双面软磁盘有两个磁头,每次读、写只选其一)、磁头运动方向、定位所需的步进脉冲、读写控制信号等;状态信息主要有索引脉冲、寻道完成、零磁道等;读、写数据是串行调制码脉冲,具体的读、写过程与硬磁盘控制器类似。

驱动器接口信号通过 50 线(用于 200 mm 软磁盘驱动器)或 34 线(用于 133 mm 或更小的软磁盘驱动器)扁平电缆传送。一个控制器可配接若干台

软磁盘驱动器,常用菊链式连接。在软磁盘驱动器上用跳线选择来使软磁盘驱动器的物理地址与程序指定的逻辑地址相对应。接口线大部分定义是标准的,但不同类型的驱动器,可能有极少数接口线定义不同。只有接口线与控制器接口线完全相同的软磁盘驱动器,才能与该控制器相连接。

早期的软磁盘控制器用逻辑元件、可编程逻辑器件或微程序控制逻辑组成。现在软磁盘控制器专用大规模集成电路芯片已广泛应用。

微型计算机的软磁盘适配器也常称为软磁盘控制器。

参考文献

刘乐善,叶济忠,胡盛斌编著.微机接口技术与应用.武汉:华中理工大学出版社,1993 (叶济忠)

ruancipan qudongqi

软磁盘驱动器(floppy disk drive, FDD) 驱动软磁盘并实施数据存取的设备。软磁盘驱动器的工作原理与基本结构类似硬磁盘驱动器,它由磁头、磁头驱动与定位机构、软磁盘引导与夹紧装置、主轴、读写电路和控制电路等部分组成。这些部件用以完成软磁盘装卸、磁盘旋转、磁头寻找磁道、读写数据和状态检测等功能。图 1 示出了软磁盘驱动器的结构框图。

工作时,软磁盘通过引导机构插入后套在主轴驱动轮上,通过关门的连锁机构动作使压紧轮将软磁盘夹紧;加载机构使磁头与软磁盘接触;主轴电机带动主轴和软磁盘旋转,达到额定速度;由步进电机、丝杠、磁头小车和零道光电传感器组成的磁头驱动与定位机构将磁头移动到零道上。这时驱动器处于就绪状态,可以接受主机来的命令。

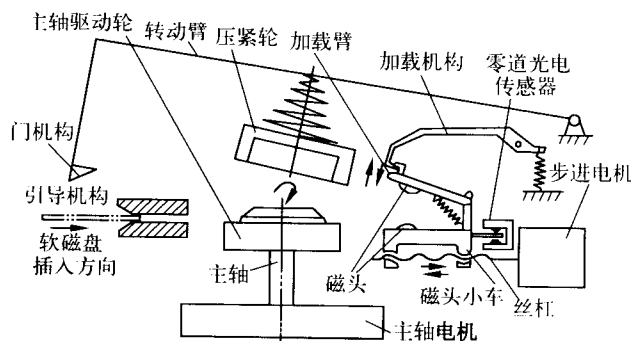


图 1 软磁盘驱动器结构框图

当软磁盘控制器接到主机命令后,经过对命令的解释,产生各种控制信号,第一步实现磁头寻道功能,使所选驱动器的磁头定位在目标磁道(即此次命令指定的磁道)上。寻道前,磁头目前所在的磁道地址已存放在道号寄存器中,目标磁道号也已存放在暂时寄存器中,比较两者求出磁头需移动的磁道数和移动方向,由控制器给出驱动步进电机的步进脉冲和方向信号,驱动磁头至目标磁道,依靠电机的电磁阻尼与传动机构的机械阻尼使磁头稳固地定位在目标磁道上。第二步检出索引,检测扇区地址,实现数据的读写功能。在读写之前,软磁盘已按规定的格式进行了初始化,因此在读写数据之前不需先检测扇区地址标志,以及读取扇区地址和校验码,经比较无误后,按所选择的操作进行读写或抹除数据,完成命令规定的要求。

纵向(水平)记录软磁盘驱动器 3.5 英寸、双面、倍密度的软磁盘存储器是常用的纵向记录驱动器的软磁盘存储器,其驱动器的技术性能如表 1 所示。

表 1 3.5 英寸软磁盘驱动器技术性能

技术性能	数 值
容量(格式化)/MB	1.44
数据传输速率/(kb/s)	500
平均寻道时间/ms	60
转速/(r/min)	300
平均旋转等待时间/ms	100
平均无故障间隔时间/h	10 000
外形尺寸	101mm × 150mm × 25mm
重量/g	500

采用嵌入伺服、直线马达驱动磁头、磁盘高速旋转的大容量软磁盘驱动器也已出现。它的道密度很高,平均寻道时间较短,数据传输率也很高,容量可达 750 MB(2003 年)。但因种种原因,诸如兼容性、接口以及在可换的软磁盘上使用嵌入伺服的不方便等,目前尚未获得普遍采用。

垂直记录软磁盘驱动器 此种驱动器在磁头、记录媒体、读写电路方面与纵向记录的软磁盘驱动器有较大的差别,其他部分基本类似。磁头与磁层的组合有三种方式:①环形头与钕铁氧体涂敷媒体组合;②单极头与 CoCr/Ni-Fe 非晶双层薄膜组合;③磁通闭合型单极头与 CoCr/Ni-Fe 非晶双层薄膜组合。采用第一种方式的软磁盘驱动器用于微型计算机。它的未格式化容量为 4 MB(格式化容量

为 2.88 MB);与普通的纵向记录软磁盘驱动器相比,其位密度高一倍,为 35 kb/i;在其他参数相同时,它的数据传输速率达 1 Mb/s。此种驱动器因使用环形头对垂直取向媒体写入,读出时垂直分量较大,信号波形呈现不对称状态,需在读电路中进行波形处理(例如使用 Hilbert 过滤器),或者在磁头前隙的后沿增设某种软磁薄膜。其他两种组合方式的软磁盘驱动器的记录密度虽然很高,但因媒体的耐磨性不佳,尚未出现成熟的产品。

参考文献

1. 张江陵,季国钧等. 外部设备设计原理. 武汉:华中理工大学出版社,1989
2. 高桥昇司著. 软磁盘机原理与应用. 李振旺,张遇吉译. 北京:电子工业出版社,1987 (张江陵)

ruancipan qudongqi ceshi

软磁盘驱动器测试(floppy disk drive testing) 通过测试仪对软磁盘驱动器机械、电气性能进行检测和评估。为保证软磁盘驱动器的质量和良好的互换性,软磁盘驱动器应在研制、生产、维护等过程中,采用各种仪器设备进行严格的检测和评估。

软磁盘驱动器机械与电路的测试和调整内容主要有:索引信号,零道信号,准备好信号,写保护信号,主轴转速,寻道,磁头加载时间,磁头定位(径向位置),磁头方位角(切向位置),磁头分辨率,磁头读出信号幅度,磁头写入波形;数据信号抖动性,内、中、外磁道信号的不对称度(测信号位漂移),内、中、外磁道读裕量(测误码率),格式化磁盘,连续读。

软磁盘驱动器测试仪可分练习仪和测试分析仪两大类。

练习仪 是一种低档的非智能或具有一定编程功能的简易型检测仪。一般在生产过程中或维修时用于对软磁盘驱动器作机械上的调整(如磁头位置调整、零道传感开关位置调整等),也可用来验证驱动器各部件的功能(如寻道、磁头定位、读写、各种状态信号控制等)。通常操作者运用练习仪和示波器、电压表及有关模拟校准盘(AAD)等即可完成一定的检测和调校工作,调整精度与操作者的技术水平有关。这种仪器结构简单,轻便,操作方便,一般每次只能测试一台软磁盘驱动器。

测试分析仪 是一类中高档的功能检测仪。其系统采用微处理器控制,可自动操作,也可手动操

作。除了练习仪的基本功能外,它还具有许多其他功能,如进行相位裕量、数据分离、信号的不对称度、磁头定位重复性、恶劣花样码读写等多种定量分析。可用来对软磁盘驱动器进行全面的检测和评估。一般情况下要与模拟校准盘及数字诊断盘(DDD)一起使用。测试程序、测量值及测试结果可在显示屏上实时显示或同时由打印机打印输出。这类仪器可同时或顺序测试 30 多台软磁盘驱动器。仪器的结构形式有便携式、台式、落地式等。

参考文献

Greenwood D. Disk-Drive Testers. Electronics Test, 1984(2): 68~81
(周学仁)

ruancipan shipeiqi

软磁盘适配器 (floppy disk adapter) 连接微型计算机总线与软磁盘驱动器,以实现微型计算机对软磁盘驱动器进行数据存取的接口部件。它既起信号适配的作用,又起控制的作用。前者指缓冲与匹配微型计算机总线与软磁盘驱动器间的传输速率,并将微型计算机总线传送的信号转换为符合软磁盘驱动器接口规约的信号;后者指执行软磁盘输入输出控制程序,将微型计算机需要存储的数据经缓冲、并串转换和编码等处理后,送往软磁盘中由程序指定的地址处进行记录;或从软磁盘中取出微型计算机需用的数据,经译码、串并转换和缓冲等处理后送往微型计算机主存;并及时检测适配器中的状态信息,供微型计算机处理。故软磁盘适配器又常称为软磁盘控制器。

软磁盘适配器与软磁盘控制器的基本功能、基本工作原理和寻道、读、写等基本控制过程是相同的,但由于它们所控制的软磁盘驱动器数量、容量、数据传输速率等差别较大,因而在具体结构、功能强弱以及驱动软件的复杂程度等方面都有所不同。微型计算机的软磁盘适配器一般控制一至两台软磁盘驱动器,置于主机箱内。早期的软磁盘适配器是一块单独的电路插卡,电路的集成度不高。后来推出软磁盘控制器专用芯片,它集成了软磁盘控制器的绝大部分功能,只需配接少量的主机接口匹配电路,即可形成完整的软磁盘适配器。常将它和硬磁盘适配器做在一块电路板上,成为软磁盘、硬磁盘适配器卡;或进一步将软、硬磁盘适配器和串、并接口电路做在一起,称为多功能卡。软磁盘适配器的典型结

构是以软磁盘控制器专用芯片为核心组成,有的专用芯片集成了全部软磁盘控制的功能,只需根据主机的要求加入少量接口匹配电路即可。这种专用芯片包括了主机接口逻辑、微程序控制逻辑、各种控制寄存器、写预补偿和数据分离电路、时钟和定时电路、驱动器接口的发送和接收驱动电路等。软磁盘控制程序可通过相应端口对有关寄存器寻址,进行读取状态、传送命令和数据等操作。控制逻辑据此形成寻道、读、写等控制信号,进行数据的串并、并串转换,检纠错处理,调制码、解编码处理等,控制被选的软磁盘驱动器一步步完成程序指定的操作。

(叶济忠)

ruanjian anquanxing

软件安全性 (software safety) 使软件所控制的系统始终处于不危及人的生命财产和生态环境的安全状态的性质。随着计算机应用范围迅速地扩大到工业控制、航天航空、医疗设备、银行金融、交通通信等领域,计算机越来越深入人们的日常生活。计算机及其软件虽然提高了人们的生活与工作条件,但也日益更加直接地关系到人们的生命、财产与人类的生存环境的安危。安全性就是计算机在这些领域的应用对软件提出的一个新的要求。

近年来,由于软件安全性问题,已经造成了一些重大事故。例如,由于控制放射性治疗设备的软件错误,在加拿大已经造成了多起癌症病人因受到过量放射性辐射而死亡。在英国伦敦,救护车调度软件刚投入使用几小时就发生故障,造成急诊病人延误达十几小时。由于软件错误,阿里亚娜-5 型火箭发射卫星失败,造成高达 25 亿美元的直接经济损失等等。根据软件故障可能造成的损失大小,IEC 国际标准 SC 65 A-123(草案)把软件危险程度分成四个层次。如果一个以计算机为基础的系统的失败会造成大量人员丧失生命,生产设备或交通设施的完全摧毁和造成巨大的经济损失,那么其危险程度是灾难性的。如果事故会造成人员丧失生命和伤亡,部分生产设备的严重损坏且造成大量的经济损失,那么其危险程度是重大的。如果错误会造成人员受伤以及一定限度内的能够盈利的生产设备和交通设施的损失,那么其危险程度是较大的。如果系统不涉及安全性问题,那么其危险程度是较小的。IEC SC 65 A-123 要求一定危险程度的软件达到一

定的可靠性,参见表1。

表1 IEC SC 65 A-123 对软件可靠性的要求

危险程度	连续控制系统	保护系统
	每小时发生危险故障的次数	请求调用时发生故障的概率
灾难性	$10^{-9} \sim 10^{-8}$	$10^{-5} \sim 10^{-4}$
重大	$10^{-8} \sim 10^{-7}$	$10^{-4} \sim 10^{-3}$
较大	$10^{-7} \sim 10^{-6}$	$10^{-3} \sim 10^{-2}$
较小	$10^{-6} \sim 10^{-5}$	$10^{-2} \sim 10^{-1}$

在要求安全性较高的领域内,计算机的应用造成灾难性事故的可能性引起了国际上对软件安全性的高度重视。人们把这类一旦发生故障就可能危及人的生命、财产和生存环境的软件称为安全第一的软件。它们的基本特征是安全性具有至关重要的意义。为了保障这类软件的质量,政府机构纷纷迅速采取法律与管理措施,一批保障软件安全性的软件生产、使用和维护的规范与标准已开始拟订,有的已经投入实施。表2给出了一些与软件安全性直接有关的标准。

表2 软件安全性相关的标准

标准文号	标准名称	发布与开发机构
Int Def Stan 00-55	The procurement of safety critical software in defense equipment—Part I : requirements; Part II : guidance	UK Ministry of Defense
Int Def Stan 00-56	Hazard analysis and safety classification of the computer and programmable electronic systems elements of defense equipment	UK Ministry of Defense
ISA-SP84(Draft)	Programmable electronic systems (PES) for use in safety application	Instrument Society of America
Mil-Std-882B Notice 1	System safety program requirements	US Department of Defense
P1228 (Draft)	Standard for software safety plans	IEEE Computer Society
RTCA /DO-178A	Software consideration in Airborne systems and equipment certification	Radio Technical Commission for Aeronautics
SC 65A /WG9 (Secrariat) 122(Draft)	Software for computers in application of industrial safety-related systems	International Electrotechnical Commission
SC 65A /WG10 (Secrariat) 123(Draft)	Functional safety of programmable electronic systems; Generic aspects	International Electrotechnical Commission
SEB 6-A	System safety engineering in software development	Electronics Industries Association
UL1998(Draft)	Standard for safety-related software	Underwriters Laboratories
N /A	Reviewer Guidance for computercontrolled medical devices undergoing 501 (K) reviewer	US food and drug administration

参考文献

1. Leveson N G. Software Safety in Embedded Computer Systems. CACM. 1991 Feb;34 ~ 46

2. Leveson N G. Software Safety: Why, What, and How, Computing Surveys. 1986. 8(2):125 ~ 163

3. Redmill F, Anderson T (eds). Safety Critical Systems, Current Issues, Techniques and Standards. Chapman & Hall, 1993

4. Wichmann B A. Software in Safety Related Systems. Wiley, 1992 (朱鸿 金凌紫)

ruanjianbao

软件包 (software package) 完成特定任务的一个程序或一组程序。可分为应用软件包及系统软件包两大类。应用软件包和特定应用领域有关。又可分为通用包及专用包两类,通用软件包根据社会的一些共同需求开发。专用软件包则大多是生产者根据用户的具体需求定制。

软件包这一术语出现于 20 世纪 60 年代。60 年代初,IBM 公司曾将 IBM 1400 系列上的应用程序库改造成更为灵活易用的软件包形式。Informatics 公司根据用户需求,以包的形式设计并开发了自动流程图生成包 AUTOFLOW。1969 年,软件开始从计算机系统中分离出来成为独立成分,软件包这个术语开始广泛使用。

软件包由一个基本配置和若干可选部件构成,既可以是源代码形式,也可以是目标码形式。用户手册或指南等文档是软件包的重要组成部分。此外,软件包的维护及技术支持也是必需的。

已成为商品的软件包数以万计。如有:TOTAL (Cincom System, Inc.), MSA Payroll Accounting System (Management Science America, Inc.), The Data Analyzer (Program Products Inc.) 等。

(梅宏 邵维忠)

ruanjian ceshi

软件测试 (software testing) 检测和评价软件以确定其质量的过程与方法,即评价软件或程序的属性和能力,以确定它是否满足所需结果的过程与方法。软件测试的通常意义是对软件进行动态评价,即用输入样例来运行程序,并比较实际输出和预期结果,但静态分析也是软件测试的组成部分。一次成功的测试是指揭示了迄今为止尚未发现的某些错误的测试,而不是指未能找出错误的测试。

软件测试可分为静态分析和动态测试。进行静态分析时,不必运行软件,只是通过对源代码进行分析,检测程序的控制流和数据流,以及发现执行不到的“死代码”、无限循环、未初始化的变量、未使用的数据、重复定义的数据等;也可能包括对多种复杂性度量值的计算。静态分析虽然不能取代动态测试,但它是动态测试开始前有用的质量检测手段。动态测试技术借助于输入样例来执行软件,一般又可分为功能测试(即黑盒测试)以及结构测试(即白盒测试)。

黑盒测试是基于软件功能的测试,它试图发现以下类型的错误:①功能不对或遗漏;②界面错误;③数据结构或外部数据库访问的错误;④性能错误;⑤初始化和终止错误。其测试用例设计依赖于软件的需求规约和设计规约,由于这种测试没有考虑程序内部代码结构,所以称为“黑盒”测试。

白盒测试在设计时考虑了程序内部代码结构,主要方法有:

(1) 基本路径测试 任何软件程序能表示成一个控制流图,程序中的如果语句或循环语句可以用带条件求值(判定)点的不同分支来表示。可以选择测试用例使得程序“从控制流图的开始到结束沿某一条执行路径”执行,下一个测试用例应选择控制流中前面至少有一条边没有走过的路径,这样的执行路径称为独立路径。程序中独立路径数目等于判定点数加 1,因此,有可能做到覆盖所有独立路径的完全覆盖测试。

(2) 语句覆盖测试 语句覆盖率是通过测试用例集所执行到的语句数目除以程序中总的可执行语句数目。通常认为,应使所测试的模块达到 100% 的语句覆盖率,除非有合理的理由不能达到。然而,即使达到 100% 语句覆盖,在软件中还是会有许多未测试到的方面。

(3) 分支或判定覆盖测试 类似于语句覆盖,但它不是统计执行过的语句数目,而是统计执行过的分支或判定的数目。一个如果语句有真、假两个分支,情况语句和循环语句可等价于分支语句。100% 的语句覆盖并不能保证 100% 的分支覆盖。

(4) 判定-条件覆盖测试 判定-条件覆盖通过执行一个判定中每个条件输出的真和假两者来达到。对于由三个比较条件组成的判定,为了达到判定-条件覆盖,用来确定整个判定的三个条件每个都需要考虑真和假的情况,其实这里有两个测试用例便可做到(例如,使三者都为真和三者都为假)。然而,即使达到判定-条件覆盖,也未必达到判定覆盖。

(5) 条件组合覆盖测试 不仅要求每个条件的求值为真和假,而且要求对所有可能的条件组合求值以达到判定条件组合覆盖。例如,在一个判定中有三个条件,则这三个条件的组合需要有 8 个测试用例。对于现代结构化编程语言,100% 的条件组合覆盖可保证 100% 的分支和语句覆盖。

软件测试策略与软件工程过程相对应:

单元测试对应程序编码,主要采用白盒测试技术。

集成测试对应软件的设计,主要检测各个单元集成过程中相互之间的接口错误以及形成新的组合后功能中的错误,多用黑盒测试技术并辅以一些白盒测试技术。

确认测试对应系统需求,以确保软件符合所有功能、行为、性能的需求规约,确认测试只使用黑盒测试技术。

此外,在软件提交给用户方时,要进行验收测试。验收测试的目的是建立系统能正常工作的信任,而不是为了发现错误。其他级别的测试还有 α 测试和 β 测试,前者是在开发者场所软件实际使用的测试;后者是软件产品正式发布前由用户方在用户场所实际使用的测试。

参考文献

1. Marciniak John J. Encyclopedia of Software Engineering. Wiley, 1994
2. Pressman Roger S. Software Engineering, A Practitioner's Approach. Fourth Edition. McGraw-Hill, 1997 (金茂忠)

ruanjian tiaoshi

软件调试 (software debugging) 发现所编写软件中的错误,确定错误的位置并加以排除,使之能由计算机或相关软件正确理解与执行的方法与过程。

软件错误可分为两种。由于表达方式不符合语法规则而产生的错误称为“语法错误”。它除了通过人工方式发现以外,通常可以由编译程序或者特定的语法检查工具检查出来。排除了语法错误的软件所表达的意义还有可能与编写者心目中的想法不一致,这种错误称为“语义错误”。它往往要通过仔细阅读源程序,或者通过“让计算机实际运行或解释该软件,并观察其实际效果”的办法才能发现。

一般来说,在进行调试工作以前,首先要发现存在着某种错误的迹象。随后的调试过程通常分为两步:①确定问题的性质并且找到该错误在软件中所处的位置;②修正这一错误。在这两步工作中,第一步的工作量最大,一旦确定了错误的性质和所在位置,排除它相对来说是比较容易的。不过要注意在修正一个错误时,要防止产生新的错误。

(周锡令)

ruanjian fangfaxue

软件方法学 (software methodology) 以软件方法为研究对象的学科。主要涉及指导软件设计的原理和原则,以及基于这些原理、原则的方法和技术。狭义的也指某种特定的软件设计指导原则和方法体系。不论何种含义,关注的中心问题是如何设计正确的软件和高效率地设计软件。

计算机硬件技术的进步,一方面为计算机在社会生活和人类活动各个方面的应用提供了物质基础,另一方面对软件也提出了越来越高的要求。软件日益增加的复杂程度促使软件方法学和软件工程的研究兴起,但是二者的侧重点不同。软件工程的研究侧重于借鉴传统工程学科,最终目的是把软件生产变成一门制造工程。而研究软件方法学的目的是寻求科学方法的指导,把软件开发活动置于坚实的基础上。但是在提出的初期,两者间的界限不是十分清楚。在以后的发展过程中,相互间的影响与渗透也处处可见。总体上讲,软件工程需要方法学的依据和指导;方法学依赖软件工程特别是环境工具来发挥实际效用。

其中程序设计方法学是最先发展起来的部分,它研究使程序设计提高质量和效率的方法和相关技术,特别着重于建立程序设计的方法学基础和各类程序设计方法。计算机科学的一些精粹分支是从此生出来的,例如,有关形式语法和编译的严格基础、程序正确性证明、程序验证、抽象数据类型、形式语义理论、结构程序设计等等。都是作为程序设计方法学的问题在20世纪60年代和70年代期间发展起来的。

软件方法学的分类和基本内容

从开发风范上看,有自顶向下的开发方法,自底向上的开发方法。在实际软件开发中,大都是两种方法的结合,只不过是应用于开发的不同阶段和以何者为主而已。

从性质上看,有形式方法与非形式方法。形式方法是一种具有坚实数学基础的方法,从而允许对系统和开发过程作严格处理和论证,适用于那些对系统安全性要求极高的软件的开发。非形式方法则不把严格性作为其主要着眼点。

从适用范围来看,有整体性方法与局部性方法。适用于软件开发全过程的是整体性方法,自顶向下方法,自底向上方法,各种软件自动化方法等均作为整体性方法。适用于开发过程具体阶段的为局部性方法,如需求分析阶段的各种需求分析方法,设计阶段

的各种设计方法。

(1) 自顶向下方法 自顶向下是一种决策的策略。软件开发涉及到决策问题:作什么决策、如何决策和决策顺序。

自顶向下方法在任何时刻所作的决定都是当时对整个设计影响最大的那些决定。如果把所有决定分组或者分级,那么决策顺序是首先作最高级的决定,然后依次地作较低级的决定。同级的决定则按照随机的顺序或者按别的方法。一个决定的级别是看它距离要达到的最终目的(软件的实际实现)的远近程度。从问题本身出发来看,或是由外(用户所见的)向内(系统的实现)看,以距离实现近的决定为低级决定,远的为高级决定。

在这个自顶向下的过程中,一个复杂的问题(任务)被分解成若干个较小较简单的问题(子任务)。并且一直继续下去,直到每个小问题(子任务)都简单到能够直接解决(实现)为止(参见**结构化程序设计**)。

(2) 自底向上方法 与自顶向下方法相反,首先作最低级的决定,其次较高级的决定,最后建立起整个系统。也就是首先对最基本的构件和系统的内部函数,然后逐步升级到有关外部函数的决策。在这个过程中,开发者手中有越来越复杂和强有力的构件可以用来构造更高级的构件,直到最后构成整个系统。

(3) 形式方法 形式方法的目的是把软件作为数学来重新发现。形式方法需要有一个健全的数学基础,典型地是由一个(建立在集合论、逻辑或是代数的基础上的)形式规约语言来给定。它提供了精确地定义以下概念的手段,如一致性、完全性、规约、实现、正确性等。它还提供手段以证明规约是可实现的,证明所建立的系统是正确地实现的,以及证明系统具有某些性质而无须通过实际运行来确定其行为。

形式方法被用来避免系统中的歧义性、不完全性、不一致性。在系统开发的早期使用形式方法有助于避免设计缺陷,否则这些缺陷会留到后来的测试、排错阶段才能发现,而造成巨大耗费。在系统开发的后面阶段用形式方法,可以帮助确定系统实现的正确性及不同实现的等价性。

(4) 软件自动化方法 是指利用计算机使软件的设计实现自动化的方法和相关技术。

长久以来,人们就希望发展出一种技术,使得计算机的用户所需要具备的机器知识和程序设计知识

尽可能地少,一项任务从提出到在计算机上最终获得解决所需人的参与尽可能地少。最理想的情况是发展出这样的软件自动化系统:它是面向最终用户的,即直接由最终用户使用;它是通用的,即对任何应用领域都适用;它是全自动的,即不需要人的干预和协助。

由于无法达到面向最终用户、通用、全自动的理想目标,一种办法是降低对一个目标的要求,以争取较好地实现其他两个目标。即只追求有限目标,使难度降低,于是有实现软件自动化的三种策略:①自底向上式策略,不追求直接面向最终用户,而是逐步向最终用户靠拢;②狭窄领域策略,不追求适用于广泛领域,而是更好地解决专门领域中的问题;③辅助式策略,不追求全自动,而是以设计者为主体,提供辅助工具。

软件自动化的实现途径。软件自动化系统从问题描述(即规约)出发,得到可执行程序,有如下四种实现途径(参见**软件自动化方法**):

过程途径,迄今最成功的途径。写一个专用的程序(编译程序或程序生成器),按部就班地进行加工以得到结果。但因为系统的每个特色都要靠相应的专门算法来实现,此途径不适于实现面向广泛领域中的最终用户的软件自动化系统。

演绎途径,或演绎综合。要得到满足某个规约的程序,相当于寻求该规约可满足性的构造性证明,并根据此证明具体构造出一个算法来。这种方法具有健全的数学基础,因此很吸引人。任何演绎方法均可用来支持软件自动化,但是在实践上还没有一种能用来证明现实规模的复杂定理。原因在于:演绎本质上是搜索推理路径的问题,而这在本质上又是指数复杂度的。不能对付复杂问题是因为搜索方法的低效,因此一个办法是让人来参预,但是这决不比程序设计更容易。另一个严重问题是,它所寻求的构造性证明,往往对应的是低效算法。

转换途径,研究得最多的途径。对软件自动化系统的输入是**甚高级语言(VHLL)**,即直接面向最终用户的非过程语言,施以一系列转换之后,得到低级的实现表示(可执行程序)。一个转换由三个部分组成:模式、条件、动作。找到符合的模式之后,校验条件,成立时施用动作。转换要求保持程序正确性不变。有两类转换:一类从高层次到低层次(垂直),将抽象变为具体;另外一类不改变抽象级别(水平),而是进行优化。转换一次次施行直到某个条件满足为止(例如找不到符合的模式)。在许多

方面转换与证明步骤并无不同,因而所面临的问题也一样。有时希望人来干预,但人也未必就有很好的办法。有少部分转换非常之关键(决策性的),其余的则不是很重要。如能使人可以干预关键部分,而其余的则自动化,当可取得更佳效果。转换方法吸引人之处在于它提供了程序设计知识的一种极清楚的表达方法。

归纳途径,将归纳推理用于程序设计。将程序的行为的实例作为规约的部分描述,通过归纳推理得出关于其行为的规律作为程序的规约,并且从它综合出程序来。相当于对实例进行推广,得到一个类,以所给出的实例为其特殊情形,所得到的程序是针对整个类的。此途径的好处是对用户不要求程序设计知识,实例非常容易理解和修改。问题是有关的技术还不能解决现实规模复杂度的程序。一种退一步的做法是不企图自动地推广实例,而是提供一个开发环境来协助用户进行推广。

参考文献

1. Gries D. Programming Methodology: A Collection of Articles by Members of IFIP WG 2.3. New York: Springer-Verlag, 1978
2. Wegner P. Research Directions in Software Technology. Cambridge: MIT Press, 1980
3. Lowry M R, McCartney R D. Automating Software Design. Menlo Park: AAAI Press /The MIT Press, 1991

(董毓美)

ruanjian fuyong

软件复用 (software reuse) 提高软件生产力和质量的一种技术,将已有软件的各种有关知识用于建立新的软件,以缩减软件开发和维护的花费。早期的软件复用主要是代码级复用,被复用的知识专指程序,后来扩大到包括领域知识、开发经验、设计决定、体系结构、需求、设计、代码和文档等一切有关方面。

发展简史

软件复用的最早例子是子程序库。但是通常认为,软件复用的正式提出是在 1968 年。D. McIlroy 在国际上首次讨论软件工程的会议上建议建立生产软组件的工厂,用产品目录上的软组件构成复杂系统,以作为解决“软件危机”的一种可能技术途径。

在 20 世纪 70 年代中开始了软件工厂的多项研究实验。80 年代软件复用技术得到美国、欧洲、日本的政府和企业界的倡导与支持,如美国先进研究

计划署 (ARPA) 的 STARS 计划;欧洲信息技术研究战略计划支持的若干计划,特别是 Eureka 软件工厂;日本的若干软件工厂 (Hitachi, Toshiba, NEC, Fujitsu 及 NTT 等);以及美国的若干公司级计划 (GTE, AT&T, IBM, Hewlett-Packard) 等等。

80 年代后期在构件库系统、构件分类技术、可复用构件的创建与分发、复用支撑环境、公司级复用计划等方面取得了许多进展。但是没有达到原来对软件复用所预期的那种在软件生产力和质量上的重大提高,据认为是因为复用的范围太狭窄,许多对效率和质量关系重大的因素未在考虑之列。这导致了 80 年代末提出对软件复用的广义理解,使得与软件计划有关的一切,包括知识,均在复用之列。这就大大拓宽了研究领域,使复用问题无处不在。

最新的研究工作涉及非技术因素:管理、经济学、文化、法律。这些问题是重要的,是可能比技术问题更难以解决的。新的复用观把这些因素也包括在内。

软件复用技术的基本内容、现状和趋势

(1) 复用什么

思想、概念、算法:被复用的是对一类问题的一般性解决方法。以算法来说,寻求算法的研究已经相当成熟。但是从复用的角度看,还需要在发展和分发标准目录方面做更多工作,并且提供专门的有关如何复用算法的详细信息的基本目录。

制成品、构件:这是一直在着重做的。80 年代初以来,软件工厂和公司级的复用计划都在从事软构件复用。面向对象的技术在构件复用上非常受重视。构件复用研究工作着重点是质量和可靠性,构件的适应性也是重要问题。特定领域的构件集合是研究的新趋势。

过程、技能:最集中的研究领域之一,即如何把软件开发过程加以形式化并加以封装,创建可复用对象的集合。技能和技术诀窍的复用主要受到专家系统研究者的关注。

(2) 复用形式及范围

垂直式:在同一应用领域中的复用,目的是得出一族系统的通用模型,可以用作样板来装配新的系统。研究重点在领域分析和领域建模,使用面向对象的术语,就是在特定问题领域中标识出一类相似系统的对象及操作和建立领域的模型。大多数软件工厂和有长期计划的机构都集中力量于垂直式复用。

水平式:目的是在不同的应用中使用通用部

件,科学子程序库、Unix 工具是这类复用的例子。主要问题在于部件的包装和表达,以及发展可供广泛接触的库和库的网络,为此需要建立部件分类标准和部件目录。

(3) 复用方式

有计划复用:系统化和正式的复用,软件工厂中的复用模式。有确定的指导原则和规程可以遵循,并且收集度量数据以评定复用的性能。这种复用需要真正来自高层的投资和承诺,而且难以预测投资的回报。还需要对现行的软件开发实践作重大的改变,对开发者有强制性的纪律要求,并且需要他们的妥协。核心问题是建立复用成熟性模型和经济学模型,以使经理人员了解预期效益、投资回报、初始代价和性能准则。

专有化复用:非正式的复用,通常也称为看机会的复用。这是现在发生得最多的情形,复用是个人行为,不是在课题一级。不存在规程,所使用的构件也不是为复用目的而专门设计的。

(4) 实现复用的途径

合成途径:用已有构件作为新系统的积木块。基于完善地建立起来的积累、高效的库系统和标准接口。虽然标榜复用所有的软件产物,但是主要着重于源代码复用。主要问题是库系统(构件选择和检索)、领域分析(构件理解和构件适应)以及接口技术(构件集成)。

生成途径:在规约级的复用。通过应用代码生成器,可以提供具有最高潜力的回报。Unix 中语法分析器的生成器 Lex 和 YACC 是著名例子。主要问题是如何形式地表达针对特定领域的规约语言、软件进程和元生成器。

(5) 单元如何复用

黑箱:复用构件不经任何修改。可复用构件是封装了的,具有标准接口。这种途径保证了更高的质量和可靠性,但是创建这种可复用黑箱的代价比较昂贵。主要问题是验证和(发给证书)证明在任何可能情形下构件都完美无缺陷地执行。

白箱:构件可以修改使得适应于具体情形的需要。这是最常见的途径,特别是在专有化复用的场合。绝大多数复用计划,包括软件工厂都属于白箱复用。主要问题是构件的参数化和内部可适应性,以及把复用扩展到更高级产物如设计和规约。

(6) 何种工作产物被复用

源代码:现状是代码复用。已有的绝大多数复用工具、环境及方法是针对代码复用,但是发展趋势

是越来越少强调代码复用。到最后,代码将从更高级的表达自动地生成。

设计:设计复用可提供比代码级复用更高的回报,但是仍是在生长中的技术。面向对象的方法可以部分地或间接地实现设计复用,但是系统化的实践仍在研究阶段。

规约:可以提供最高的回报,是生成式复用的焦点。当规约可供复用时,通常也就实现了设计级和代码级的复用。

对象:越来越受到重视的复用物。有多种方法、工具和环境来创建对象,面向对象的方法看来是未来的复用技术。趋势是走向集成化的工具和方法来覆盖开发全过程。

正文:即除代码之外的所有工作产物,特别是供人使用的文档。正文复用日益重要,趋势是把可复用正文与所有其他工作产物一起集成。

体系结构:最大的复用单位。要分析应用领域以找出共同的设计,然后用作基本样板来集成可复用部件或是开发专门的代码生成器。领域分析同时支持生成式和合成式复用。

参考文献

1. McIlroy D. Mass-produced Software Components. In: Buxton J M et al. Software Engineering Concepts and Techniques. 1968 NATO Conference of Software Engineering. New York: Petrocelli /Charter, 1969. 88 ~ 98
2. Biggerstaff T J, Perlis A J. Software Reusability, Volume I. Concepts and Models, Volume II. Applications and Experiences. New York: ACM Press, 1989
3. Freeman P (ed). Software Reusability. New York: IEEE Computer Society Press, 1987

(董樵美)

ruanjian gongcheng

软件工程 (software engineering) 应用计算机科学理论和技术以及工程管理原则和方法,按预算和进度实现满足用户要求的软件产品的工程,或以此为研究对象的学科。

发展简史

术语“软件工程”第一次出现在 1968 年的 NATO 会议上。这次会议以及而后的一些会议集中讨论了大型软件开发项目中出现的诸如软件质量、开发成本以及按期交付等问题,以着手解决软件开

发失控的所谓“软件危机”,并提出了一些克服“软件危机”的策略。

20世纪60年代末至80年代初,围绕软件项目,开展了有关软件开发风范、开发方法以及支持工具的研究。其主要成果是,提出了**瀑布模型**、**演化模型**、**螺旋模型**以及**结构化程序设计方法**等,开发了一些结构化程序设计语言(例如**PASCAL语言**、**Ada语言**)、结构化软件开发方法和支持工具。并且,围绕项目管理,出现了一些管理方法和相应的支持工具。

随着软件系统规模的增大、复杂性的提高以及在关键领域应用的开展,20世纪80年代以来,人们更加关注软件生产技术的研究和实践,并注重软件工程管理和软件质量的研究。其主要成果是,开发了具有广泛应用的**面向对象方法**和相关的语言,提出了软件过程概念以及**能力成熟度模型(CMM)**,开展了**计算机辅助软件工程(CASE)**的研究与实践,特别是在**软件复用**方面取得了有效成果,如构件模型、复用机制等。

基 本 内 容

软件工程的框架 概括为:目标、活动和原则。

软件工程项目 生产具有正确性、可用性和开销合宜的产品。正确性是指软件产品达到预期功能的程度。可用性是指软件基本结构、实现以及文档为用户可用的程度。开销合宜是指软件开发、运行的整个开销满足用户要求的程度。

软件开发活动 生产一个最终满足需求且达到工程目标的软件产品所需要的活动。软件开发的基本活动包括:需求分析、设计、实现、验证与确认和维护。

(1) 需求分析是在一个抽象层上建立系统模型的活动。这一活动产生了需求规约,用以作为软件开发人员和客户之间契约的基础,并作为以后开发阶段的输入。

(2) 设计定义了实现需求规约所需的结构。包括软件体系结构(数据和程序结构),以及详细的处理算法,即所谓**设计规约**,给出了实现软件需求的软件解决方案。

(3) 实现是由设计规约到代码的转换。为此,需要选择特定的语言和工具。

(4) 验证与确认是一项评估活动,其中主要包括需求规约、设计规约以及实现代码的评估。目前开展较多的是实现代码的评估,一般包括程序的单元测试、集成测试和系统测试等。软件测试技术主

要包括基于代码结构的白盒测试和基于规约的黑盒测试。验证与确认这一项评估可以是动态的或是静态的。在动态评估中,以选定的输入来执行程序或程序段,并与预期结果进行比较。静态评估是不执行程序的分析,例如模型评审、代码“走查”以及程序的形式化验证等。

(5) 维护是在软件发布之后所进行的开发或修改,包括对发现错误的修正以及对环境的变化所进行的必要调整等。

软件工程原则 围绕软件开发,提出了以下基本原则。

(1) 选取适宜的开发风范 在系统设计中,经常需要权衡软件需求、硬件需求以及其他因素之间的相互制约和影响,适应需求的易变性,因此,要选用适宜的开发风范,以保证软件开发的可持续性,并使最终的软件产品满足客户的要求。

(2) 采用合适的设计方法 在软件设计中,通常需要考虑软件的模块化、信息隐蔽、局部化、一致性以及适应性问题,因此,采用合适的设计方法,支持这些问题的解决和实现,以达到软件工程的目标。

(3) 提供高质量的工程支持 软件工程如其他工程一样,需要提供高质量的工程支持,例如配置管理、质量保证等,才能按期交付高质量的软件产品。

(4) 有效的软件工程管理 软件工程管理直接影响可用资源的有效利用,以提高软件组织的生产能力。因此,只有对软件过程实施有效管理时,才能实现有效的软件工程。

软件工程研究内容 主要包括:软件开发过程,软件开发方法,软件工程管理,软件质量特征,软件过程度量以及计算机辅助软件工程(CASE)工具、环境等。

软件开发过程 将用户需求转换为软件系统(产品)的一组有序的活动。其中,每一活动是由一组任务定义的,而任务是把输入转换为输出的一组操作。

在整个软件生存周期中,包含了三类软件过程,它们是基本过程,支持过程和组织过程。基本过程是软件开发人员所从事的一切活动。支持过程是软件需求方和软件开发方各类支持人员所从事的一切活动。组织过程是管理人员所从事的一切活动。

每项软件工程通过“**剪裁过程**”可定义应遵循的一组有序的规程和活动。尽管在实践中,每项软件工程都有自己特定的软件开发过程,但软件开发

风范主要有五种,它们是:迭代风范,也称为演化风范,转换风范,螺旋风范,瀑布风范,以及第四代风范。

迭代风范是一种有弹性的过程模式,该模式由一些小的开发步组成,每一步历经需求分析、设计、实现和验证,产生软件产品的一个增量。通过这些迭代,完成最终软件产品的开发。其中,一次迭代所产生的增量产品,往往发布给软件客户,以获取他们的反馈,用于指导相继的步骤。

转换风范的过程模式基于待开发系统的形式化需求规约。在此基础上,通过一系列转换,将这一形式化的需求规约转化为它的实现。

第四代风范的过程模式是围绕特定语言和工具而设计的。这一风范的过程依据高层描述,自动生成代码。

关于螺旋过程模型和瀑布过程模型,可参见**螺旋模型**和**瀑布模型**。

以前,软件工程所关注的是按进度测量过程中每一活动的结果,得到一个产品;而现在则转向关注软件开发过程,并提出了一系列过程描述的技术,支持软件项目和开发者发现适宜的过程。

软件开发方法 为了实现系统的分析和设计,软件开发过程通常遵循特定的途径和方向。典型的途径有:**结构化方法**、**面向数据结构方法**、**面向对象方法**以及**维也纳开发方法(VDM)**等。过程的方向确定了用于创建模型和设计解的特定的抽象层次。软件开发方法给出了指导软件开发活动的基本原则、技术和步骤。

(1) **结构化方法** 一种系统化的软件系统建模方法。包括结构化分析和结构化设计。结构化分析方法以分层的数据流图和控制流图工具,开发系统的功能模型和数据模型。著名的有 E. Yourdon 和 T. DeMarco 的结构化分析(SSA)。结构化设计包括软件体系结构设计、过程(功能)设计和数据设计。体系结构设计以系统的功能模型为基础,逐层精化,最终形成系统的模块(子系统)以及它们之间的控制关系。其中,强调模块(子系统)的高内聚和低耦合。过程(功能)设计针对体系结构中的每一模块,给出它的过程属性描述,即算法设计。数据设计将系统的数据模型转换为数据结构。

(2) **面向数据结构方法** 结构化方法的变形,其中,着重**数据结构**而不是数据流。这类方法的共同特征是:①标识关键的信息对象和操作;②以顺序、选择、重复三种基本结构层次式地表达数据结

构;③提供一组步骤,把层次式的数据结构映射入程序结构。这类方法的范例有**数据结构化系统开发方法(DSSD)**或**Warnier/Orr方法**、**Jackson系统开发(JSD)方法**。

(3) **面向对象方法** 一种以对象、对象关系等来构造软件系统模型的系统化方法。包括面向对象分析和设计。其中,对象由封装的属性和操作组成;对象类是具有相似属性、操作和关系的一组对象的描述。类是系统建模、系统设计以及实现等活动可复用的基础。这类方法著名的有:G. Booch 的 OOD, J. Rumbaugh 的 OMT 以及 P. Coad 和 E. Yourdon 的 OOA, OOD 等。目前,对象管理组织(OMG)发布的**统一建模语言(UML)**以及统一软件开发过程(USDP)受到了业界和学术界广泛关注,特别是 UML 以及相应的支持工具已在软件开发中得到了广泛的应用。

软件工程管理 一项软件工程主要涉及五大要素:人员、进度、质量、成本和实现的需求。因此,管理的责任主要是围绕这五大要素进行“规划和组织”、“领导和控制”以及评估等。

在软件开发中,为了及时提供项目状态反馈信息——这是项目成功的关键。经常使用:①**软件度量学**,测量软件产品(包括文档)和软件过程的不同属性;②**成本估算度量学**,依据系统的规模和功能的测量、预测和调度资源,包括整个产品(系统)开发中的人力资源;③**系统化的测量技术**,评估实现组织目标和项目目标的进展等。

在软件工程管理中,一种广泛使用的策略是“走查”,以后发展为“审查”,即建立一个由各方人员(但不包含代表管理的人员)组成的工作小组,通过复审事前的工作产品,发现其中的错误。

软件质量特征 **软件质量**是软件产品和服务能够表明满足客户要求的全部性质和特征。软件质量可以通过一组属性予以测量,这组属性是高质量软件的特征,例如正确性、功效性、方便性、易维护性、可测性、鲁棒性以及可用性等。关于软件质量属性的需求是一些非功能需求,是需求规约的组成部分。在软件工程实践中,质量的每一个属性,并不总是可以度量的,但必须进行与之相关的一些方面的度量。

可以说,没有一个项目可以实现全部的质量特征。其中重要的原因是:①受有限的时间和资金资源的限制;②一些质量特征与其他特征之间是相互冲突的,例如,对一个特定的环境,一个系统很难既十分方便又具有最大功效。

在一个大型的软件系统开发中,为了使最终的软件产品具有客户所期望的质量属性,关键的途径是与客户进行交流,其中最有效的技术是原型构造。

所有软件工程活动都应该有质量保证,作为该活动的一个目标。对于软件工程所选择的过程,要重视所预期的质量特征;实施软件工程活动要确保实现所预期的质量特征;并选择一种度量方法对确定的质量属性进行测量。

能力成熟度模型(CMM)与计算机辅助软件工程(CASE) 自1968年软件工程概念提出以来,历经近40年的实践,人们发现,对于软件开发还需要更加灵活的、但又规范的途径,包括贯穿软件产品整个开发过程的反馈和适当的调整。为此,有关政府和软件产业界共同努力,研究并开发了实现高质量大型软件系统的方法和工具。其中具有影响的工作是CMM的提出和CASE工具、环境的研制。

CMM为开发组织改善其开发过程提供了一个框架。该框架将成熟度分为五个等级,它们是初始级、可重复级、已定义级、已管理级和优化级。

CMM为评估一个软件开发组织的能力给出了一种方法。

CMM的提出引发了大量的过程改善活动。并举办了多次区域性和国际性会议,讨论采用CMM的经验和过程改善问题。在CMM概念的基础上还提出了个人软件过程(PSP)等。

可用的CASE工具和环境的出现是软件工程实践进步的最好体现。

CASE工具大体上可分为两类:一类是支持工程管理活动的工具,例如**配置管理工具**,成本估算工具,调度工具以及文档工具等;另一类是支持开发活动的工具,包括**设计工具**(如原型速成工具,建模工具等),**程序设计辅助工具**(如**调试工具**,代码生成器等),**测试工具**以及**维护工具**。

CASE环境是CASE工具的集成,并协调地工作于同一机器环境中,支持整个软件开发工作。通常,CASE环境包括配置管理工具,以跟踪软件开发过程中的中间产品。

展 望

伴随着软件工程领域的发展,出现了一系列新的问题,并成为新的研究热点。例如,①对依赖性进行了一定的扩展,不仅包括可靠性,还包括软件系统的安全操作等问题;②**软件复用**已成为一个重要的质量特征,很多项目希望购置一些可复用的构件来建造软件系统;③正在开发一些逆向工程技术,支

持老系统的维护,其中这些系统或者没有文档,或者与当前演化的系统不相匹配。

虽然软件工程的研究迄今已经有了很大进展。但遗憾的是,软件工程的原理还未能充分影响许多软件工程的实践。并且,尽管有关各个方面(政府有关部门,软件企业,专家学者等)都做了很大努力,但在研究环境中提出的新技术到工业实践的转换仍然是很缓慢的。

在软件工程领域中,目前仍存在很多挑战。例如,形式化的分析技术,过程和质量特征的测量技术以及分析和测量工具等。另外,软件系统与其他工程系统之间的集成,也是软件工程领域中的一项十分困难的问题。随着软件工程的不断发展,相信可以逐步解决以上问题并迎接新的挑战。

参考文献

1. 杨芙清. 杨芙清文集. 北京: 北京大学出版社, 1998
2. Ralston A et al. Encyclopedia of Computer Science(Forth edition). New York: Van Nostrand Reinhold, 2000
3. Marciniak J J. Encyclopedia of software Engineering. New York: John Wiley & Sons Inc., 1994

(杨芙清)

ruanjian gongcheng huanjing

软件工程环境 (software engineering environment) 以软件工程为依据,支持大型软件生产的系统。简称SEE。

SEE是软件工程学科诸方面研究发展的物化表现,具有以下特点:

(1) SEE强调支持软件生产的全过程。实践表明:软件工具仅能支持软件生命周期中某些特定活动;工具箱将一些相互关联的工具组织到一起,使其相互通信,从而支持了生命周期中某些成组活动,自动化程度有所提高;软件开发环境旨在通过环境信息库和消息通信机制实现工具的集成,从而为软件生命周期中某些过程的自动化提供了更有效的支持。不同的过程模型导致不同类型的**软件开发环境**。然而,软件过程领域的最新研究结果表明软件过程概念已不仅局限在软件开发和软件维护,而是发展到包括系统集成和软件产品的制作与生产(参见**软件过程**)。SEE旨在解决软件过程中的各个过程和活动如何按照各条路径并行完成。管理、支持、获取、供应等过程贯穿于整个生存周期,特别是管理

活动和支持活动,从合同观点来看,对于软件生产具有更为重要的地位。因此,促使人们在更高的层次上考虑如何运用“整体大于各部分之和”的系统工程观点,提高整个软件过程的整体计算机辅助支持程度。

(2) SEE 强调大型软件的工业化生产。软件发展已经形成了大规模的产品市场。因此,软件产品的生产也必将走上工业化的道路。所谓工业化是指能够形成规模经济的生产形式。工程化注重软件生产过程中软件生产者具有一定的工程训练,遵循一定的工程准则;工业化则更加强调软件生产过程中生产资料的作用,表现为把大量的生产者的技能和经验转移到生产工具之中,从而减少生产过程中人工的劳动比例,增大软件产品中生产工具的增值比例。

软件生产能否走向工业化与软件技术和软件工程技术的发展水平密切相关。软件范型的研究、第四代语言、面向对象构件库等技术的发展为软件生产走向工业化提供了有效的途径。SEE 则是为软件工业化生产提供一整套的支持设施。然而,目前软件工业化生产程度还较低,有待于进一步提高。

(3) SEE 以集成和剪裁作为主要技术途径,实现软件工业化生产的目标。SEE 具有多维特性,表现在不仅要集成与软件开发技术相关的工具,还要集成与支持技术、管理技术相关的工具,并将它们有机地结合在一起。

(4) 标准化。软件生产走向工业化需要建立相应的工业标准。近年来,美国国家标准局 NBS 以及电气和电子工程师学会(IEEE)的有关部门开始围绕 SEE 研究和建立相应的标准,试图规范 SEE 所涉及的过程、活动、要素等,并比照 ISO 的开放系统互连(OSI)参考模型提出 SEE 参考模型,这是值得我们注意的,预示着真正的软件工业化生产时代的到来。

(陈钟)

ruanjian gongcheng jingjixue

软件工程经济学 (software engineering economics) 从经济学的观点来研究、分析如何有效地开发、发布软件产品和支持用户使用这一产品的学科。

软件工程的目的是通过有关学科的应用使人们开发出来的软件系统成为对用户有用的产品。然而软件工程的效果是否良好不仅取决于计算机科学、软件方法学以及软件工具与环境的使用,也要看它

是否满足经济学和社会效益方面的考虑。

软件工程经济学研究的问题包括:

(1) 成本估算技术与成本估算模型的建立和使用

成本估算是软件工程经济学的中心问题。成本估算的经典方法是基于 B. W. Boehm 在 70 年代提出的结构化的成本模型(COCOMO)。使用 COCOMO 模型时的主要问题是:需要对待开发的软件的大小(源程序行数)事先有所估计。进行这一估计需要其他方面的知识,例如依靠专家或拥有这类知识的专家系统,因而是 COCOMO 模型本身无法解决的。此外,由于近年来面向对象技术的广泛使用和软件部件可重用程度的提高,软件开发过程中新编写源程序的比重日益减少,而对日益庞大的部件库的内容进行搜索、了解所花工作量的比重日益增加,COCOMO 模型的这一缺点就更加明显,需要加以改进。

(2) 软件工程中不同决策的“成本-效益”分析

完成一个工程项目常有多种途径,项目负责人面对不同的方案需要对每种方案所需花费的成本和可能获得的效益进行分析和比较,以便选出最佳方案。

(3) 多目标决策分析

从经济学的观点对软件工程中要达到的多个(有时是互相冲突的)目标进行协调、作出决策以及对这些同时要达到的目标进行管理的技术。

(4) 不确定性的处理和风险分析

当我们利用某种模型或理论进行分析或预测时,不管这模型或理论多么完善,我们都需要有一些信息作为分析的出发点。然而在实际情况下,我们往往无法获得完整的信息,在这种情况下,分析的结果便带有不确定性,从而据此作出的决策便带有风险。软件工程经济学在这种场合中就要研究如下的一类问题:①分析比较按不同主观估计作出的决策之后所带来的风险的大小和性质;②研究是否值得为了获得更多的信息以降低风险而要多付出一些代价。

(5) 工期估计和控制

能否按时完成工程项目与缩短工期也是软件开发中经常面临的问题,因此工期估计和控制也是软件工程经济学要研究的重要课题。

参考文献

巴里·W. 贝姆著. 软件工程经济学. 北京:中国铁道出版社,1990

(周锡令)

ruanjian gongju

软件工具 (software tool) 一类软件,用来辅助计算机软件的开发、运行、维护、管理、支持等过程中的活动或任务。使用软件工具能节省软件生产开销,提高软件生产率和产品质量。

早期人们构造软件,从本质上讲就是进行程序设计。引导程序、装入程序和编辑程序可看作是最早使用的软件工具。在汇编语言和高级程序设计语言出现以后,与之相应的汇编程序、解释程序、编译程序、连接程序和排错程序就成了当时用于开发软件的主要工具。

以后,编辑程序从行编辑发展到全屏幕编辑,进而又产生了可以识别语言文法结构的语法制导结构化编辑程序。

20 世纪 60 年代末出现了软件工程以后,支持软件需求分析、设计、编码、测试、维护和管理等活动的各种先进工具相继诞生,其中有的已能实现程序和文档的自动或半自动生成以及程序正确性的形式化验证。

进入 80 年代后,随着交互式图形技术的发展,出现了用户界面工具(如窗口系统)。近年来,新颖的多媒体软件工具也应运而生。

由于软件开发过程本身是由若干活动构成的,所以人们很自然地提出了把支持特定开发过程的单个工具集成在一起的要求。**工具集成**意味着把若干工具或工具片段结合起来,使几个相关的工具能协同操作。80 年代中期,把一组软件工具按一定的软件开发方法和过程模型有机地组织起来的集成化软件开发环境已开始出现。

软件工具的种类繁多,可以从不同的观点来进行分类。由于大多数软件工具仅限于支持软件生存周期过程中的某些特定活动,所以通常把它们分成需求分析工具、概要设计工具、详细设计工具、编码工具、测试工具和维护理解工具等。

软件管理和支持过程往往要跨越多个活动,支持这些活动的常用工具有:软件项目管理工具、软件配置管理工具、软件质量工具以及文档、表格工具等。

以下列举几种典型的软件工具:

需求分析和概要设计工具 支持需求分析活动的主要工具有:数据流图(DFD 图)工具、实体-关系模型工具、状态转换图工具、面向对象的模型工具、原型工具以及数据字典工具等。支持概要设计活动的主要工具有:分析和验证需求定义规约的工具、

程序结构图(SC 图)设计工具和面向对象的设计工具等。

这些工具用于辅助建立软件的系统模型。如数据和控制流的表示、数据内容(数据字典)定义、处理过程表示、规约和其他各种模型表示,还可能通过一致性和确认检查,提供若干级别的分析结果,以防止把错误传播到设计或实现阶段。

目前,多数分析和概要设计工具能支持结构化分析和结构化设计(SA/SD)方法。这些工具把特定的表示法、层次式的分析和设计以及从分析到设计的转换结合起来,最后形成软件可加工

的表示。

详细设计和编码工具 支持详细设计的工具有 HIPO 图(层次输入-处理-输出图)工具、PDL(设计程序语言)或 PAD(问题分析图)工具以及代码转换工具等;支持编码活动的工具有正文编辑程序、语法制导结构化编辑程序、编译程序、汇编程序、连接程序和符号调试程序等。第四代语言、应用程序生成程序和面向对象程序设计环境等可看作是新一代的编程工具。

第四代语言、代码生成程序已经改变了传统的系统开发方式,它们能比常规程序设计语言在更高层次上抽象说明一个应用信息系统,不仅能把系统描述自动转换成程序,而且能帮助校验系统规约的正确性,使其输出结果与用户需求相符合。

面向对象编程环境往往和特定的语言相关(如 Visual C++, Smalltalk, Eiffel 等)。典型的面向对象编程环境与新型的用户界面相结合,具有许多特定功能。如通过“浏览程序”在包含数百上千个软构件的对象库中查找,以确定是否有在当前应用系统中可复用的对象。

测试工具 支持测试活动的主要工具包括:静态分析程序、动态覆盖率测试程序、测试结果分析程序、测试报告生成程序、测试用例生成程序和测试管理工具等。

静态分析程序通过对源程序的程序结构、数据流和控制流进行分析,可发现一些隐藏的错误。在并行(并发)程序设计里,能辅助检测死锁和同步异常等情况。

覆盖率测试程序通过对程序的执行流进行探测,得到语句、分支和路径覆盖情况(执行次数),测试有关变量值的断言。

维护和理解工具 支持软件维护和理解活动的主要工具有程序结构分析程序、文档分析工具、程序

理解工具以及源程序至 PAD 图或流程图的自动转换工具等。它们可分为逆向工程工具和再次工程工具两类。

逆向工程工具对已经开发完成的源程序进行分析,抽取程序的系统结构、控制结构、数据结构和数据流等信息,生成结构化图形方式的分析和设计模型以及其他设计信息。用图形表示分析结果的静态逆向工具已被称为“代码可视化工具”。还有一类用来监控软件执行,并利用监控期间获取的信息来建立程序行为模型的动态逆向工程工具。

再次工程工具用来支持重构一个功能和性能更为完善的改进的软件系统。如代码重构工具能重新构造程序代码,使之与结构化程序设计要求相符;数据再次工程工具能交互修改数据库的逻辑结构,并重构一个新的数据库物理设计。

项目管理工具 通常项目管理工具把重点放在一个特定管理环节上,而不提供对管理活动包罗万象的支持。工具能对软件项目的工作量、成本和工期进行估算,可定义工作分解结构、工作调度计划以及对项目开发状况进行连续的跟踪和控制。此外,还能用工具对软件项目进行度量,提供软件生产率 and 软件产品质量的指标。

配置管理工具 能辅助完成软件配置项的标识、版本控制、变化控制、审计和状态统计等基本任务;使各配置项的存取、修改和系统生成易于实现,从而能简化审计过程,改进状态统计,减少错误,提高系统质量。

评价软件工具并不在于它的功能如何齐全,而在于它是否能提高软件的开发效率和质量。好的软件工具除了便于使用、工作可靠、性能良好和技术服务支持完善以外,还要考虑工具应能剪裁和定制,以适应环境变化以及特定用户的要求。此外,工具应易于安装到用户已有的环境和系统中,并与其他工具和数据库相匹配。

参考文献

1. Chikofsky E. Computer-Aided Software Engineering (CASE). 2nd ed. Los Alamitos: IEEE Computer Society Press, 1993
2. Fisher A S. CASE-Using Software Development Tools. New York: John Wiley & Sons Inc., 1988
3. Gane C. Computer-Aided Software Engineering: The Methodologies, the Products, and the future. Englewood Cliffs: Prentice Hall Inc., 1990 (金茂忠)

ruanjian goujian

软件构件 (software component) 软件系统中具有相对独立功能,可以明确辨识,接口由规约指定,与语境有明显依赖关系,可独立部署,且多由第三方提供的可组装软件实体。软件构件须承载有用的功能,并遵循某种构件模型。可复用构件是指具有可复用价值的构件。

当前,对软件构件的定义及其内涵尚未形成一致共识,不存在公认的构件模型。然而,一个基本的认识是:构件的接口规约中,除其功能规约外,还需包括其对外交互连接的描述和支持自省的自描述信息。现有的构件模型基本可分为 3 类:

(1) 构件描述-分类模型 描述构件的所有对用户查找、理解、选择、适应性修改及使用构件有帮助的信息,以及所有对构件库管理者分类和管理构件及构件间关系有帮助的信息。其目的是使得构件易于在构件库中被有效、高效地分类、储存和检索,易于被用户理解和复用,这就是构件库的数据模型。代表性的工作有(可复用库互操作组织, RIG)提出的统一数据模型(UDM)和基本互操作性数据模型(BIDM)。

(2) 构件规约-组装模型 描述构件的功能和行为规约(包括构件对外提供的功能和需要外界提供的功能)、构件应用的语境以及构件间的交互和组装。用于规约构件并在设计层次上组装构件。这类模型通常体现于接口定义语言(IDL)、构件描述语言(CDL)和软件体系结构描述语言(ADL)中,基本都遵循概念、内容和语境(3C)模型。

(3) 构件实现模型 描述构件在源程序级或二进制目标码级的实现机制,用于指导人们以某种程序设计语言或以某种可执行单元的形式来实现构件,也称基础设施模型。这类模型并不关心构件的内部实现方式,重点是构件的接口封装机制,不过,当前主流的构件实现模型均是基于面向对象技术扩展而来。典型的代表性工作有:微软公司的 COM/DCOM, OMG 的 CORBA/CCM, SUN 公司的 JavaBeans 和 Enterprise JavaBeans。通常,构件实现模型是和支持构件运行的软件中间件平台紧密相关的。

由于大多数构件模型都是基于面向对象的概念,有时人们将构件简单地用作对象的同义词。然而,构件和对象是两个相对独立的概念。对象是封装了状态和行为并有独特标识的软件实体,通过类定义对象的行为和结构。而构件除了封装状态和行

为外,至少还需提供其对外交互的连接点和供自省的接口,并可同时拥有多个接口。构件由构件类型或构件模板定义。在使用上,对象(类)可在语言层次上通过继承等白盒方式被使用,而构件的实现通常完全对外界隐蔽并且有时只是以二进制代码形式存在,只能采用黑盒使用方式。实际上,一个构件可以由一个或多个对象实现,甚至可以由非面向对象语言的传统方法来实现,只需对外提供符合构件实现模型的接口即可。

参考文献

1. Crnkovic I and Larsson M. Building Reliable Component-Based Systems. Artech House, July 2002
2. Szyperski C. Component Software: Beyond Object-Oriented Programming. New York: Addison Wesley / ACM Press, 1998
3. Heineman G T and Councill W T. Component-Based Software Engineering: Putting the Pieces Together. Reading, Massachusetts: Addison-Wesley, 2001

(梅宏 谢涛)

ruanjian goujian ku

软件构件库 (software component library)

储存可复用软件构件、相关软件资产及其他有关属性信息,用以支持开发人员共享构件资源的软件包。构件库是实施成功软件复用的重要基础设施。可复用的软件资产包括需求规约、软件设计规约、源代码、目标码、测试计划、测试用例以及用户指南等对新的软件开发有用的资源。

软件构件库的研究和实践主要涉及以下几个方面:

构件分类和检索机制

构件的分类和检索机制是软件构件入库和检索的基础。一方面,拥有大量可复用构件的组织必须以一种易于分类管理而又方便复用者检索的机制来表示和保存构件资产;另一方面,有效的构件检索机制能够降低构件查找和理解的成本,而构件的合理表示和分类正是实现高效方便的检索的基础。

目前有很多构件分类和检索方法,从构件的表示出发可以分为人工智能方法、超文本方法和信息科学方法三类;而根据复杂度和检索效果的不同则可以分为基于文本的、基于词法描述的和基于规约的编码和检索。信息科学方法是实际复用项目中应用较为成功的一类,并且以枚举、刻面、属性值、关键词和正文检索几种方法较为常见,其中刻面分

类方法能够表达丰富的构件信息,尤其为人关注。刻面分类方法将关键词(术语)置于一定的语境中,并从反映构件本质特性的不同视面(刻面)将构件分类。

构件入库

软件构件库应对每个推荐入库的构件进行定性和定量的评价,比较复用该构件所需的代价和所获的收益,作为对构件分级的主要标准。构件在入库时应提供以下内容:①复用者手册,提供有关构件特性、安装、验证及操作的完整指令;②构件摘要信息;③构件分类信息;④构件实体(代码或文档)或其存放位置(在某些情形下,构件的实体也可以不存放在软件构件库中);⑤构件测试计划、目标和预期结果。

构件配置管理

软件构件库应当维护构件自身的版本信息和构件间相互的关联和配置关系,通过构件配置为复用者提供更全面的服务。同时,还需维护复用者对构件的查询、提取、使用和反馈信息,并以此来改进构件库的查询机制。

构件库工具

构件库工具分为管理工具和用户工具两类。管理工具包括:支持构件的分类;支持对分类机制的维护;生成事务和状态报告;支持对构件和分类词汇的配置管理;支持对问题报告的追踪。用户工具为复用者提供有效的构件查询手段。

构件库及其标准化

软件构件库作为管理软件构件、促进软件复用的核心机制,是面向复用的开发过程和基于复用的开发过程的中介和衔接。国际上已经建立了若干大型构件库系统,也先后推出了定位各异的构件管理工具。基于此,构件库的标准化工作也在展开,如 STARS 项目考虑了开放体系结构的构件库之间共享资源和无缝互操作的问题,提交了“开放体系结构的构件库框架(ALOAF)”;复用库互操作组织(RIG)致力于开发构件库互操作的解决方案,提出了统一数据模型(UDM)和基本互操作数据模型(BIDM);北大西洋公约组织(NATO)针对 NATO、NATO 参与国和项目承包商制定了一组软件复用标准,包括“可复用构件开发标准”、“可复用软件构件库管理标准”和“软件复用过程标准”。

为了充分利用软件构件资源,推行基于构件的软件开发方法,美国还进一步提出了公共国家级软件构件库的建设设想。

参考文献

1. 杨芙清, 梅宏, 李克勤. 软件复用与软件构件技术. 电子学报, 1999, 27(2): 68 ~ 75
2. RIG Uniform Data Model for Reuse Libraries (UDM). Reuse Library Interoperability Group, RPS-0002, January 1994
3. RIG Basic Interoperability Data Model (BIDM). Reuse Library Interoperability Group, RPS-0001, April 1993, revised: January 1995 (孙艳春)

ruanjian guocheng

软件过程 (software process) 软件生存周期中的一系列相关过程。又称软件生存周期过程。过程是活动的集合, 活动是任务的集合, 任务要起到把输入加工成输出的作用。活动的执行可以是顺序的、重复的(迭代的)、并行的、嵌套的, 或者是有条件地引发的。

细言之, 软件过程有三层含义: 一为个体含义, 即指软件产品或系统在生存周期中的某一类活动的集合, 如软件开发过程, 软件管理过程等等; 二为整体含义, 即指软件产品或系统在所有上述含义下的软件过程的总体; 三为工程含义, 即指解决软件过程的工程, 它应用软件工程的原则、方法来构造软件过程模型, 并结合软件产品的具体要求进行实例化, 以及在用户环境的运作, 以此进一步提高软件生产率, 降低成本。

软件过程的工程含义体现在如下几个方面:

(1) 软件过程的概念已不仅仅局限于软件开发和维护工作, 它已发展为系统的集成和软件产品的制作与生产。这是由于软件复用技术已经较为成熟, 软件产品市场已具较大规模, 对系统集成和软件产品供应有迫切需要。为此提出了获取过程和供应过程, 从而反映了软件过程不仅要有工程视面, 也要有合同视面(包括系统视面和用户视面)。

(2) 提高生产率和软件质量, 其关键在于管理和支持能力。所以软件过程又特别重视管理活动和支持活动, 提出了管理过程和支持过程, 从而反映了软件过程中的管理视面。

(3) 由于区分了软件开发环境和业务运作环境, 并且系统或软件产品也不一定全部自行开发, 可从获取过程得到所需的软件, 从而提出了运作过程, 反映了软件过程中的运作视面。

(4) 软件过程研究的对象已扩展到从事软件活动的人之工作。不同的角色, 其视面不同, 所负责的

软件过程亦不相同。如获取者或供应者, 按他们的合同视面只负责获取过程或供应过程; 管理者按其管理视面负责的是管理过程; 用户和操作人员按运作视面负责的是运作过程; 开发人员和维护人员按其工程视面负责的是开发过程和维护过程; 介入支持过程的人员按他们支持的目标负责支持过程的某些工作。

软件过程中的各个过程和活动, 是按照几个线条并行地完成的, 不仅开发、运作和维护贯穿整个生存周期, 而且管理、支持、获取、供应等过程也贯穿整个生存周期。

软件过程有各种分类方法。按性质分有基本过程、支持过程和组织过程; 按特征分有管理过程、开发过程和综合过程; 按照不同人员的工作内容来分有**管理过程**、**获取过程**、**供应过程**、**开发过程**、**运作过程**、**维护过程**、**支持过程**。不管哪种分类方法, 把软件过程运用到具体机构和具体应用领域或具体项目时, 会把各种过程和活动进行剪裁, 从而形成本机构或具体项目的软件生存周期过程模型, 这就是**剪裁过程**, 另外, 还可能涉及到基础设施过程、改进过程和培训过程。

管理过程 软件生存周期中管理者所负责的一系列活动。负责对所从事的过程, 例如获取、供应、开发和支持等过程的活动和任务进行管理; 软件管理过程适用于必须对自己的过程进行管理的任何一方。

获取过程 获取者获得一个系统或软件产品的一系列活动和任务。它从确定获取该系统或软件产品的需求开始, 经过招标准备、合同准备、谈判及修改、对供应方的监督等活动, 直至验收完成方告结束。获得该系统或软件产品的机构可就部分或全部获取活动与某机构签订合同, 后者根据获取过程开展相应的活动。二者皆可称作获取者。

供应过程 供应者为获取者提供软件产品的一系列活动。它从理解系统或软件产品的需求开始, 经过准备投标、签订合同、制定计划、实施和控制、评审和评价等活动, 直至交付完成。供应者是那些提供软件产品的机构。

开发过程 软件开发人员所负责的一系列活动, 其目的是依据合同成功地开发并交付软件。当要开发新的系统, 或对已有的系统进行版本升级以及对已有系统进行有开发活动的移植时, 都要涉及开发过程。

运作过程 用户和操作人员用户的业务运作

环境中为了使系统或软件产品投入运行所进行的一系列有关的活动。此过程包括对系统或软件产品的运作活动和用户运作时对他的支持活动。此过程的目的是在软件开发过程完成后,将该系统从开发的环境转移到用户的业务运作环境中运作;在运用中对用户的要求提供帮助和咨询;并对运行效果作出评价。

维护过程 软件维护人员所负责的一系列活动,目的是在保持软件整体性能的同时修改它,使它达到某一需求,直到其退役才告终止。从维护方式上讲有三种维护:改正维护、适应维护以及改善维护。当软件由于错误、缺陷、问题需要改进和修改,以及对相应文档进行修改时,都要涉及维护过程。

支持过程 有关各方按他们支持的目标负责的一系列相关活动。支持过程有助于系统或软件产品的质量,有助于它们的顺利运作。这类软件可以被其他类软件过程或本类中的其他软件过程所使用。软件过程的各个活动均可使用支持过程。支持过程可由使用它们的机构来实施;或作为一种服务,由一个独立的机构来实施;也可作为项目的一项规定内容,由客户来实施。一个支持过程中的活动,由实施该过程的机构负责。这类软件过程包括:文档过程、配置管理过程、质量保证过程、验证过程、确认过程、联合评审过程、审计过程、问题解决过程等。

剪裁过程 对软件过程和活动实施剪裁的过程。将一选定模型以及相关标准应用于某一领域或具体软件项目,形成该领域的模型及标准或该软件项目的各个软件过程和活动。最初选定的模型是相对抽象的,具有相对普遍性的,而所选标准是描述活动全集的,将它们针对某领域剪裁意味着形成该领域的相对特殊的模型及标准;将它们针对软件项目进行剪裁意味着形成该项目的软件过程和活动,这是剪裁过程的双重意义。

基础设施过程 建立和维护任何其他过程所需的基础设施的过程。基础设施可以包括硬件、软件、工具、技术、标准和开发、运行、维护的设施。软件过程中的许多过程都应当明确该过程的基础设施。定义并建立所需的基础设施,并在其他相关过程执行时维护所建立的基础设施,是本过程的主要活动。

改进过程 建立、评估、度量、控制和改进软件生存周期的过程。主要活动为:制定一套组织计划,评价相关过程并实施分析、改进活动。软件过程中的任何一个过程都可能涉及此过程。

培训过程 为系统或软件产品提供人员培训的

过程。软件的获取、开发、操作或维护的效果主要取决于有关人员所具备的知识和熟练程度。因此,拟定人员培训计划并及早实施是非常必要的。本过程要完成的主要活动有:制定所需的人员计划及培训计划,开发培训资料以及实施培训活动等。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes — IEEE Std. 1074 ~ 1991
2. ISO /IEC 12207:1995 Information Technology — Software — Part 1: Software Life-Cycle Process
3. システム開発取引の共通フレーム (SLCP-JCF94) (朱三元)

ruanjian guocheng moxing

软件过程模型 (software process model) 对软件过程的结构和属性的抽象描述。它通常描述了软件过程的一定抽象层次和看待软件过程的一种特定观点。软件过程的描述可以是形式化的、半形式化的或非形式化的。

软件过程模型不仅针对某个特定的软件项目,而是概括了一类软件过程的共同结构和属性,所以它具有普遍性。由于它描述了某一类软件过程,从而有别于其他类软件过程的结构和属性,所以它又具有特殊性。

软件过程模型应该描述某一类软件项目的软件开发活动的一切重要的过程细节,应具有可操作性。传统的软件开发模型(如软件生命周期模型)的抽象层次太高,仅指明软件开发活动的范围和顺序,没有对过程的细节描述以及过程步骤的明细分解,从而造成基于这些模型的软件过程对软件项目的计划、管理、质量控制等缺少描述,软件项目的成功往往在很大程度上取决于项目参加人员的素质和经验,无疑严重制约了软件产品的开发和生产。

软件过程模型包括活动模型、角色模型、产品模型、资源模型和约束模型等子模型。这些子模型分别抽象描述软件过程的基本成分的类型、结构和属性。

软件过程模型与过程建模密不可分。**过程建模**是通过软件过程定义和软件过程设计而建立软件过程模型的活动。一方面软件过程模型的类型、特性直接影响过程建模方法,而另一方面软件过程模型是过程建模的直接产物。

为了使一个软件过程模型有效、精确地应用于具体的软件项目,需要根据该项目的目标对软件过

程模型进行剪裁,并按照该项目的各种参数进行例化。由此获得一份详细的该项目所特有的软件过程描述(即软件过程实例),同时也考察了软件过程模型的普遍性和可复用性。通过过程模拟活动,虚拟执行该过程实例,从而以反馈信息为依据,既可以改进和完善过程实例,又可改进和优化软件过程模型。将过程实例投入实际的执行和运行,使该项目进入实际运作过程,在监控、管理和支持等活动的同时,收集有关的反馈信息,再次改进和优化过程实例和软件过程模型,这称作过程改进。

参考文献

1. 柳军飞,唐稚松. 软件过程建模语言研究. 软件学报,1996,8: 449~457
2. 朱三元,钱乐秋,宿为民. 软件工程技术概论. 北京: 科学出版社,2002
3. 周伯生,徐红,张莉. 过程工程原理与过程工程环境引论. 软件学报,1997,8(增): 519~534

(朱三元)

ruanjian kaifa fangfa

软件开发方法 (software development method)

软件开发过程所遵循的办法和步骤。软件开发活动的目的是有效地得到一些工作产物,也就是一个运行的系统及其支持文档,并且满足有关的质量要求。软件开发是一种非常复杂的脑力劳动,所以经常更多讨论的是软件开发方法学,指的是规则、方法和工具的集成,既支持开发,也支持以后的演化过程(交付运行后,系统还会变化:或是为了改错,或是为了功能的增减)。

关于组成软件开发和系统演化的活动有着各种模型(参见软件生存周期,软件开发模型,软件过程),但是典型地都包含了以下的过程或活动:分析、设计、实现、确认(测试验收)、演化(维护)。

有些软件开发方法是专门针对某一开发阶段的,属于局部性的软件开发方法。特别是软件开发的实践表明,在开发的早期阶段多做努力,在后来的测试和维护阶段就会使费用较大地得以缩减。因此,针对分析和设计阶段的软件开发方法特别受到重视。其他阶段的方法,从程序设计发展的初期起就是研究的重点,已经发展得比较成熟(参见程序设计,维护过程)。除了分阶段的局部性软件开发方法之外,还有覆盖开发全过程的全局性方法,尤为软件开发方法学注意的重点。

对软件开发方法的一般要求

当提出一种软件开发方法学时,应该考虑许多因素,包括:①覆盖开发全过程,并且便于在各阶段间的过渡;②便于在开发各阶段中有关人之间的通信;③支持有效的解决问题的技术;④支持系统设计和开发的各种不同途径;⑤在开发过程中支持软件正确性的校验和验证;⑥便于在系统需求中列入设计、实现和性能的约束;⑦支持设计师和其他技术人员的智力劳动;⑧在系统的整个生存周期都支持它的演化;⑨受自动化工具的支持。此外,在开发的所有阶段有关的软件产物都应该是可见和可控的;软件开发方法应该可教学,可转移;还应该是开放的,即可以容纳新的技术、管理方法和新工具,并且与已有的标准相适应。

当评价一种具体的软件开发方法时主要看四方面的特征。①技术特征:即支持各种技术概念的方法特色。如层次性、界面、控制流、数据抽象、过程抽象、并行性、安全性、正确性等。②使用特征:即用于具体开发情形时的有关特色。如可理解性、可转移性、可复用性、自动化工具的支持、生存周期的范围、任务范围、使用的广度、阶段过渡的易行性、对正确性的支持、可重复性、产品易修改性。③管理特征:即增强对软件开发活动管理的能力方面的特色。如可管理性、支持或是阻碍集体工作的程度、中间阶段的确定、工作产物、配置管理、阶段结束准则、计划性、费用估计等。④经济特征:即给软件开发机构产生的在质量和生产力方面的可见效益。如分阶段的局部效益、全生存周期效益、获得此方法的代价、使用它的代价、管理的代价等。

在一切方面都好的方法并不存在,也没有一种方法能适应于所有软件的开发需要。当需要选用一种软件开发方法时,可考虑以下的因素:①对该特定的软件开发方法是否已经具有经验,或者有受过训练的人员;②开发班子的组成情况;③为开发工作提供的环境如何;④任务计划管理的组织结构如何;⑤要解决的问题的领域性质;⑥开发工作时间进度框架;⑦是否有适当的自动化工具。

分析阶段使用的软件开发方法

需求分析阶段的任务是把软件的功用范围的一般性陈述精化为一个具体的规约,作为以后全部活动的根据。规约要表达出系统接受和产生什么数据、执行什么功能、确定了什么接口、施加了什么约束。也就是对问题及其解决的需求建立一个模型。模型主要刻画系统功能即“做什么”的问题,它在一

定程度上还刻画软件结构,即由该软件的组成成分构成软件的方法和表示(参见**软件结构**)。

主要的方法有

(1) **结构化分析** 使用得最广的需求建模方法,以数据流图和控制流图为基础,系统分析员划分出流变换函数,其次用状态迁移图来创建行为模型,用数据词典开发成数据模型。结构化分析最初是针对普通的数据处理应用发展起来的,起初是作为人工纸上作业的方法,以后发展为有自动化工具的支持。著名的有 E. Yourdon 和 T. DeMarco 的结构化系统分析(SSA),C. P. Gane 和 T. Sarson 的信息系统结构化分析设计及实现(STRADIS)。以后又发展出可用于实时系统开发的 P. T. Ward 和 S. J. Mellor 的软件工程需求分析(SERA)等方法。有许多软件工具支持结构化分析,以创建模型和帮助保证一致性和正确性(参见**结构化方法**)。

(2) **面向数据结构方法** 结构化方法的变种之一,着重于数据结构而不是数据流。这类方法的共同特征是:①协助系统分析员标识关键的信息对象和操作;②信息结构是层次式的;③数据结构的表达要求用顺序、选择、重复等合成构造;④提供一套步骤以把层次式数据结构映射入程序结构。这类方法的例子有数据结构化系统开发(DSSD)或 Warnier/Orr 方法, JSD 方法(参见 **Jackson 系统开发方法**)。

(3) **面向对象分析(OOA)和数据建模** 需求分析的面向对象方法是通过把类、对象、属性、操作作为最基本的构件来构造问题的模型。面向对象观点把对象的分类、属性继承、消息通信都组合在模型中。对象模型可以把问题的任何方面都标识为对象,特别是数据和进程。加工操作是对象的一部分,可通过向对象传送一个消息而启动。一旦定义了一个类,它就形成建模、设计、实现等各个级别的可复用性的基础,从一个类中可以实例化一个新对象。OOA 的主要目的就是标识出对象的类来(参见**面向对象方法**)。

数据建模可以看成是 OOA 的特例。它用实体-联系图作为主要的表达手段,数据建模着重在定义数据对象(不封装加工的操作),以及它们相互关联的方式。数据建模用于数据密集型的应用,也可以用作结构化分析的补充记法。

设计阶段使用的软件开发方法

设计阶段的任务是把需求翻译成软件的某种表示形式。它接受的是组成需求的信息模型、功能模型和行为模型,而产生数据设计、体系结构设计和过

程设计作为工作结果。数据设计把信息模型转换成数据结构,体系结构设计定义程序的主要结构构件之间的关系,过程设计把结构构件转换为软件的过程描述。

从项目管理的观点,软件设计可以分为两步。初步设计专注于从需求到数据及软件体系结构的转换,详细设计专注于体系结构表达的求精,以便得到详细的数据结构和软件的算法表达。

主要的方法有结构化方法,面向数据结构的方法,面向对象的方法等。许多方法不仅仅用于需求分析阶段,而且也覆盖了以后的设计阶段。如属于结构化方法的结构化分析与设计技术(SADT),系统化活动建模方法(SAMM),W. P. Stevens, G. J. Myers 和 L. L. Constantine 的结构化设计(SD)等;属于面向数据结构方法的 JSD 方法,Warnier/Orr 方法(数据结构化系统开发 DSSD),程序的逻辑构造方法 LCP 等;面向对象的方法更是一种全局性方法,即覆盖了从分析、设计直到实现的开发方法。

形式化开发方法也是全局性方法中的重要一类,具有坚实的数学基础,被看成是开发正确的软件(安全性第一的系统)的有效方法。例如 VDM 方法和 Z 等(参见**形式方法**)。

尽可能地复用已有软件的各种有关知识于新软件开发活动的各个阶段,也是十分受重视的一种方法(参见**软件复用**)。

除此以外,许多新应用还涉及专门的用户界面设计活动,以建立人机交互机制。用户界面是进入交互式软件应用的门径,界面的面貌涉及多方面因素,因此用户界面设计是一个反复进行的过程。也就是建立一个设计模型,作为原型予以实现,交由用户校验,根据他们的意见进行修改。如此继续,直到满意为止。为此有许多界面设计和原型化工具,一般也称为用户界面工具集,或是用户界面开发系统。这些工具提供模块或对象,以便于创建诸如窗口、选单、设备交互、出错消息、命令,以及一个交互式环境的许多其他元素(参见**用户界面、用户界面管理系统**)。

参考文献

1. Birrell N D, Ould M A. A Practical Handbook for Software Development. Cambridge: Cambridge University Press, 1985

2. Fisher A S. CASE Using Software Development Tools. 2nd Edition. New York: John Wiley & Sons, 1991

(董槿美)

ruanjian kaifa huanjing

软件开发环境 (software development environment) 支持软件产品开发的软件系统,简称 SDE。它由**软件工具**和环境集成机制构成,前者用以支持软件开发的相关过程、活动和任务,后者为工具集成和软件的开发、维护及管理提供统一的支持。

软件开发环境的发展大体可以划分为三个阶段。

第一阶段 软件开发环境的前身是软件工具。随着软件工程的兴起,20 世纪 70 年代中期开始出现了支持程序开发、维护的工具。这些工具大都将重点放在建立程序的相关文档上,其基本出发点是将文档建立过程也作为系统开发过程,而不是在系统开发结束后再补写文档,从而保证了文档的可信度。其中典型的工具是 ISDOS 系统中的 PSL/PSA,它主要用于系统报表及文档的生成。随后,工具箱的思想开始出现。在 70 年代末期开始使用“环境”这一术语,其中最典型的是 Xerox 公司的面向对象语言 Smalltalk 程序设计环境。这一阶段软件开发环境尚处于萌芽状态。

第二阶段 80 年代起,“程序设计环境”、“软件开发环境”等有关环境的术语广泛使用,环境的研究成了热点。这段时期,支持图形设计方式的第二代软件工具开始大量涌现,包括支持结构化分析设计方法的工具,如支持数据流图、模块结构图、状态变迁图等的编辑及分析的工具。第二代软件工具的特性是:①对结构化方法的自动化支持;②对单个系统分析员的支持;③对软件开发过程的部分覆盖;④对分析效率及确认能力的改善。同时,集成这些工具为一体的软件开发环境也得到发展,出现了以环境信息库为核心的软件开发环境。典型的环境有:Excelsator, IEF, IEW, AD/cycle, Software Backplane, STP。这些环境的显著特点有:①采用环境信息库,工具围绕信息库集成;②支持软件开发模型及软件开发方法,例如瀑布模型和结构化方法;③集成机制的研究有了较大的发展,出现了集成型软件开发环境。80 年代后期, NIST/ECMA 提出了集成化环境参考模型(图 1)。欧洲信息技术研究战略计划(ESPRIT)的 PCTE 采用了这个参考模型,1990 年成为欧洲计算机制造商协会(ECMA)的标准。但软件开发环境的国际标准尚未形成。

第三阶段 90 年代开始出现支持面向对象方法与技术的软件开发环境。受到学术界和产业界的重视。

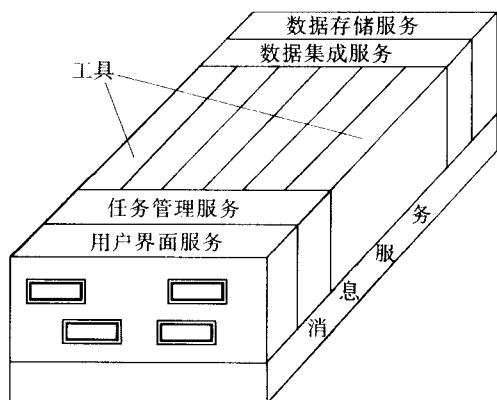


图 1 集成化环境参考模型

软件开发环境可按以下几种角度分类:

按软件开发模型及开发方法分类,有支持**瀑布模型**、**演化模型**、**螺旋模型**、**喷泉模型**以及结构化方法、信息模型方法、面向对象方法等不同模型及方法的软件开发环境。

按功能及结构特点分类,有单体型、协同型、分散型和并发型等多种类型的软件开发环境。

按应用范围分类,有通用型和专用型软件开发环境。其中专用型软件开发环境与应用领域有关,故又可称为应用型软件开发环境。

按开发阶段分类,有前端开发环境(支持系统规划、分析、设计等阶段的活动)、后端开发环境(支持编程、测试等阶段的活动)、软件维护环境和逆向工程环境等。此类环境往往可通过对功能较全的环境进行剪裁而得到。

软件开发环境由工具集和集成机制两部分构成,工具和集成机制间的关系犹如“插件”和“插槽”间的关系。

工具集 软件开发环境中的工具可包括:支持特定过程模型和开发方法的工具,如支持瀑布模型及数据流方法的分析工具、设计工具、编码工具、测试工具、维护工具,支持面向对象方法的 OOA 工具、OOD 工具和 OOP 工具等;独立于模型和方法的工具,如界面辅助生成工具和文档出版工具;亦可包括管理类工具和针对特定领域的应用类工具。

集成机制 对工具的集成及用户软件的开发、维护及管理提供统一的支持。按功能可划分为环境

信息库、过程控制及消息服务器、环境用户界面三个部分。

环境信息库 是软件开发环境的核心,用以储存与系统开发有关的信息并支持信息的交流与共享。库中储存两类信息,一类是开发过程中产生的有关被开发系统的信息,如分析文档、设计文档、测试报告等;另一类是环境提供的支持信息,如文档模板、系统配置、过程模型、可复用构件等。

过程控制和消息服务器 是实现过程集成及控制集成的基础。**过程集成**是按照具体软件开发过程的要求进行工具的选择与组合,**控制集成**实行工具之间的通信和协同工作。

环境用户界面 包括环境总界面和由它实行统一控制的各环境部件及工具的界面。统一的、具有一致视感(Look & Feel)的用户界面是软件开发环境的重要特征,是充分发挥环境的优越性、高效地使用工具并减轻用户的学习负担的保证。

较完善的软件开发环境通常具有如下功能:

- (1) 软件开发的一致性 & 完整性维护。
- (2) 配置管理及版本控制。
- (3) 数据的多种表示形式及其在不同形式之间自动转换。
- (4) 信息的自动检索及更新。
- (5) 项目控制和管理。

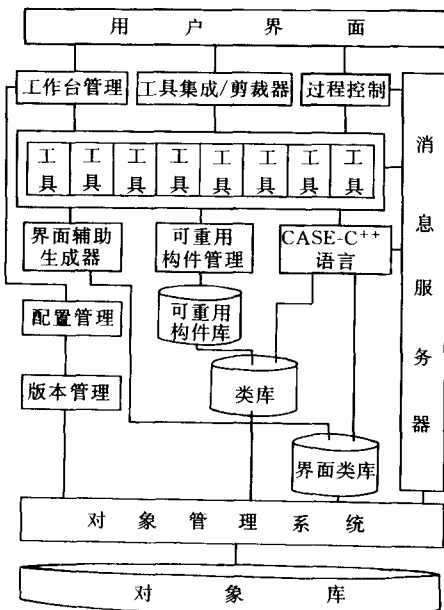


图2 一个软件开发环境实例

(6) 对方法学的支持。

图2是我国在“八五”科技攻关中研制的一个典型的软件开发环境。

当前,对软件开发环境的研究与实践仍在不断深入。预计新一代的软件开发环境将在以下几个方面取得进展:

- (1) 环境功能的智能化。
- (2) 软件开发过程的可视化。
- (3) 形成统一的国际标准。

参考文献

1. Chikofsky E. Computer-Aided Software Engineering (CASE). 2nd ed. Los Alamitos: IEEE Computer Society Press, 1993
2. Endres A and Weber H. Software Engineering Environments and CASE Technology. Berlin: Springer-Verlag, 1991
3. Norman R and Minder C. Integrated CASE. Theme issue of IEEE Software. Los Alamitos: IEEE Computer Society, Mar, 1992
4. 杨芙清,邵维忠,梅宏. 面向对象的 CASE 环境青鸟 II 型系统的设计与实现. 中国科学(A 辑), 1995, 25(5): 533 ~ 542 (杨芙清)

ruanjian kaifa moxing

软件开发模型 (software development model)

软件开发全部过程、活动和任务的结构框架。软件开发包括需求、设计、编码和测试等阶段,有时也包括维护阶段。

软件开发模型能清晰、直观地表达软件开发全过程,明确规定了要完成的主要活动和任务,用来作为软件项目工作的基础。对于不同的应用系统、采用不同的开发手段和方法,使用各种不同的程序设计语言以及各种不同技能的人员参与工作,它还应允许采用不同的软件工具或各种不同的软件工程环境。模型都应该是稳定有效和普遍适用的。

最早出现的软件开发模型是 1970 年 W. Royce 提出的瀑布模型。该模型给出了固定的顺序,将生存期活动从上一阶段向下一阶段逐级过渡,如同流水下泻,最终得到所开发的软件产品,投入使用。但实践表明,各个阶段间的关系并非如此简单。由于阶段评审可能出现向前阶段的反馈,致使在各阶段间产生环路,瀑布流水出现上流。瀑布模型为软件

开发与维护提供了一种有效的管理模式,根据这一模式制订开发计划、进行成本预算、组织开发人员,以阶段评审和文档控制为手段有效地对整个开发过程进行指导,从而保证了软件产品的质量。瀑布模型 20 多年来之所以广为流行,是因为它在支持开发结构化软件、控制软件的开发复杂度、促进软件开发工程化方面起着显著作用。与此同时,瀑布模型在大量软件开发实践中也逐渐暴露出它的缺点。其中最为突出的缺点是该模型缺乏灵活性,无法通过开发活动澄清本来不够确切的软件需求。这些问题可能导致开发出的软件并不是用户真正需要的软件,并且这一点在开发过程完成后才有所察觉。面对这些情况,无疑要进行返工或是不得不在维护中纠正需求的偏差。但无论上述哪种情况都必须付出高额代价,并将为软件开发带来不必要的损失。另一方面,随着软件开发项目规模的日益庞大,由于该模型不够灵活等缺点引发的上述问题显得更为严重。

为弥补瀑布模型的不足,近年来已经提出了多种其他模型。常见的有:

演化模型 软件开发实践表明,许多开发项目由于人们对软件需求的认识模糊,很难一次开发成功。因而,返工再开发难以避免,常常要作两次开发,其产品才能令用户满意。第一次作试验开发,其目标只是在于探索可行性,弄清需求,第二次则在此基础上获得较为满意的产品。通常称第一次试验性产品为“原型”。演化模型在克服瀑布模型缺点、减少由于软件需求不明确给开发工作带来风险方面,确有显著效果。软件系统的原型有多种形式:

(1) 丢弃型——原型开发后,已获取了更为清晰的需求信息,原型无需保留而废弃;

(2) 演示型——开发原型仅以演示为目标;

(3) 样品型——原型规模与最终产品相同,只是原型仅供研究用;

(4) 增长式演化型——原型作为软件最终产品的一部分,可满足用户的部分需求,进一步在此基础上开发,则可增加需求,实现后再次交付使用;

(5) 简陋型——用较短时间开发的简易原型。

螺旋模型 将瀑布模型与演化模型相结合,并且增加了两者所忽略的风险分析。螺旋模型通常用以指导大型软件项目的开发,它将软件项目开发分别划分为制订计划、风险分析、实施开发以及客户评估四类活动。沿着螺旋线每旋转一圈,表示开发出一个更为完善的新软件版本。如果开发风险过大,开发

者和客户无法承受,项目有可能因此终止。多数情况下会沿着螺旋线继续下去,自内向外逐步延伸,最终得到满意的软件。

喷泉模型 喷泉一词本身体现了迭代和无间隙特性。系统某个部分常常重复工作多次,相关功能在每次迭代中随之加入演进的系统。所谓无间隙指在开发活动,即分析、设计和编码之间无明显边界。

智能模型 也称为基于知识的软件开发模型,它综合了上述若干模型,并得到专家系统的支持。该模型应用基于规则的系统,采用归约和推理机制,帮助软件人员完成开发工作,并使维护在系统规约一级进行。为此,开辟了知识库,将模型本身、软件工程知识与特定领域的知识分别存入数据库。以软件工程知识为基础的生成规则构成的专家系统与含有应用领域知识规则的其他专家系统相结合,构成了这一应用领域软件的开发系统。

参考文献

1. John J Marciniak. Encyclopedia of Software Engineering. John Wiley & Sons Inc., 1994
2. John A Mc Dermid. Software Engineer's Reference Book. Butterworth-Heinemann Ltd., 1991

(郑人杰)

ruanjian kekaoxing

软件可靠性 (software reliability) 在规定运行环境下、规定时间内,软件无失效运行的概率。所谓失效,指发生了造成软件背离所期望输出的错误故障。软件可靠性可以用可靠性函数 $R_T(t)$ (软件在 $[0, t]$ 时间内不发生失效的概率) 描述,也可以用平均无故障时间 (MTTF)、失效强度函数等描述。与软件正确性验证互为补充,软件可靠性度量成为评价软件正确性的一种重要途径。

软件可靠性度量有两类基本方式:可靠性估计和可靠性预测。可靠性估计根据系统测试或系统运行时的失效数据建立模型,通过统计推理技术得到软件可靠性;可靠性预测根据软件(失效)数据和度量预计将来的软件可靠性。软件可靠性模型基于错误引入、错误消除和运行环境等影响软件失效的主要因素,给出软件失效过程的数学描述,其作用是根据软件的失效历史来估计和预测软件可靠性。软件可靠性模型根据其可靠性表达是否依赖于程序中剩余错误数,可分为错误计数模型(例如,一般泊松模型)和非错误计数模型(例如, Musa-Okumoto 对数模型)。

软件可靠性的早期研究主要集中在根据不同的假设提出不同的可靠性模型,结果出现了大量的模型。当今的共识是没有一个模型能适用于所有软件。因此,研究热点转向对不同的情形用组合模型的方法来更精确地估计可靠性,在软件开发早期预测可靠性等。同时,软件可靠性可以用来指导整个**软件生存周期**的软件开发和项目管理,以此为基础形成了以软件可靠性为目标的工程化实践——软件可靠性工程。

参考文献

1. Lyu M R (ed.). Handbook of Software Reliability Engineering. McGraw-Hill and IEEE Computer Society Press, 1996
2. Lyu M R. Software Reliability Theory, Encyclopedia of Software Engineering. Wiley, 2002
3. Bastani F B and Ramamoorthy C V. Software Reliability, Encyclopedia of Computer Science, 4th edition. Macmillan Reference Ltd., 2000 (王戟)

ruanjianku

软件库 (software library) 用于特定应用领域的软件的集合。

软件库中的所有程序必须具有相互的一致性及相容性,这些一致性和相容性主要体现在它们的对外接口和内部设计上。具体而言,软件库应具有如下基本特征:①所有程序提供类似的用户接口;②所有程序具有固定及相同的文档格式;③程序间易于相互组合以求解大型问题;④所有程序均在共同的低级系统设施上构建;⑤所有程序具有相同的编码及移植标准。

早在 20 世纪 50 年代初期,英国就出现了用于数值计算的软件库(当时称为程序库)。早期的软件库均用于数值计算并用机器语言或汇编语言编写。60 年代起,为了更好地帮助用户及促进销售,大量计算机制造商开始致力于软件库的研制。同时,众多大学、科研机构及私人企业也开始感到研制有用的软件库的迫切需要,软件库的应用领域也开始从单纯的数值计算拓广到统计分析、商业事务等领域。大多数程序采用高级语言(如 FORTRAN, COBOL 等)编写。

软件库的研制一直是人们关注的焦点。特别是在数值计算、统计分析及图形图像处理等领域市场需求很大。70 年代和 80 年代,软件库的发展尤其迅速,其编程语言也开始包括 Ada, PASCAL 及 C

等。既有对已有软件库的改进和维护,也有新软件库的不断推出。

在软件库的开发和维护中,如下问题必须予以重视:

(1) 易移植性 要求软件库能适用于不同的硬件平台。早期大量的用 FORTRAN 书写的软件库在易移植方面均有较大困难,通常是对不同硬件平台提供不同版本。其主要原因是 FORTRAN 在不同计算机上的方言化和不同机器间的算法差异。现在已有很大改善,一方面是由于语言标准的制定及采用,另一方面,是由于采用了编译预处理及运行库等技术,解决了机器间的算法差异。

(2) 错误处理 为防止用户程序失败及编程错误,软件库需具有较强的错误检测能力,诸如对输入参数的合法性及计算过程的有效性等检查。为了使用户能够更好地处理出错情况,软件库应具备良好的错误处理机制。

(3) 存储分配 许多程序的工作空间决定于其参数,为了减轻用户在存储分配上的负担,软件库中具有某种形式的自动存储分配能力是非常有用的。

(4) 文档 完备易读的文档是软件库能否被很好地使用的关键因素之一。现今的软件库除了提供详细的用户手册之外,还具有方便用户交互地获取使用信息的联机求助功能。

对软件库的研制仍在继续,仍有许多工作要做。如新的应用领域的开拓、旧软件库性能的改善、对新的计算机体系的适应等,特别地,在语言的采用上,C 语言已成为主要的候选语言。(梅宏 邵维忠)

ruanjian lijie

软件理解 (software understanding; software comprehension) 通过数据收集和分析来了解(特别是在设计文档不完全的情况下)软件的功能及其实现机理的方法和过程。

软件理解是软件维护中的重要工作。软件投入使用后,往往需要不断演进,以修正错误、扩充功能和适应新环境。对现有软件的理解是进行这一工作的前提。

软件理解一般有四项任务:识别程序单位、跟踪控制流、跟踪数据流以及综合程序逻辑。具体可以分为四级:

(1) 实现级 分析检查单个的程序结构,程序典型地表示成抽象语法树(AST)、符号表或普通源正文。

(2) 结构级 分析检查程序结构过程中的结构关系,明确表示程序组成部分之间的依赖关系。

(3) 功能级 分析检查程序结构和行为(功能)之间的关系,同时研究完成程序结构的合理性。

(4) 应用级 检查特定于应用领域的概念。

软件理解是一项复杂任务,需要借助工具来进行,可用的工具种类很多,参见理解工具。

参考文献

李必信,郑国梁,李宣东等. 软件理解研究与进展. 计算机研究与发展,1999,36(8) (郑国梁)

ruanjian liushui

软件流水 (software pipelining) 一种开发循环程序指令级并行性的方法,其基本思想是:在资源限制、数据相关、控制相关和周期性条件等的保证下,一个循环可以等价变形,使循环的一次执行可以在前一次执行结束前启动,从而使多个循环体按照流水线方式并行执行,因此称为软件流水。

在一般程序中,循环占据了绝大部分的处理机执行时间,因此,缩短循环程序的执行时间非常重要。软件流水能够消除循环程序中的绝大多数数据相关,大幅度提高循环程序的指令级并行性,充分利用硬件资源,加速循环程序的执行。

美国的 J. A. Fisher 于 1981 年提出了第一个软件流水算法,称为路径调度法。从那之后的二十多年来,世界上出现了很多种软件流水算法,归纳起

来,可以分为两大类:模调度法和核心识别法。在模调度法中,首先根据数据相关和资源限制等条件确定循环体启动间距,然后按照这个固定的间距顺序展开循环体,最后收拢形成装入部分、排空部分和新循环体。核心识别法需要重复展开并调度多个循环体,直到发现重复模式为止,Fisher 提出的路径调度法属于核心识别法。

图 1(a)是一简单的 C 语言程序,循环体内共有 6 个操作。从图 1(b)中可以看出,由于所有相邻的操作之间都存在有读写数据相关,所以,6 个操作只能串行执行。假设每个操作的执行时间均为一个时钟周期,则执行一个循环体需要 6 个时钟周期。图 1(c)是采用模调度法的软件流水结果,假设处理机内有加法器、乘法器和存储器部件各一个,他们能够独立并行工作,延迟时间均为一个时钟周期。由于在操作 3 与操作 4 之间存在一个强连通块,即同一个循环体内的操作 3 与操作 4 之间存在有有关 t3 的读写数据相关,相邻两个循环体的操作 4 与操作 3 之间存在有有关 t4 的读写数据相关,从而形成一个封闭的读写相关回路,因此,启动间距为 2。从图 1(c)中看出,开头 4 个时钟周期为软件流水的装入部分,从第 5 个时钟周期开始,直到第 196 个时钟周期为止是新循环体,这时,加法器、乘法器和访问存储器部件同时工作,每两个时钟周期就能够执行完成一个循环体。最后还要附加 4 个时钟周期的排空部分。

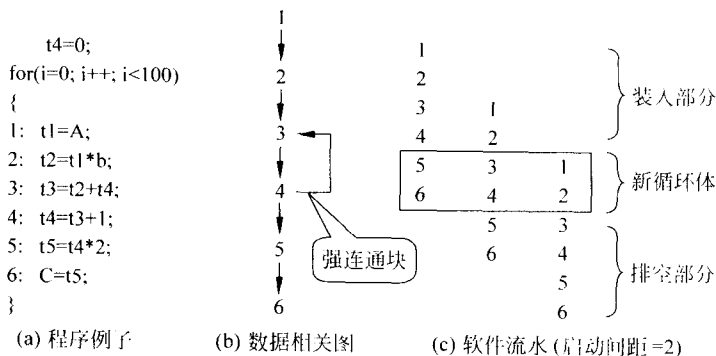


图 1 软件流水的例子

对于基本块(循环次数确定,单入口单出口,循环体内没有条件分支、调用和中断等操作),目前的许多软件流水方法都能够得到最优或接近最优的结果。当循环体内包含有条件分支操作时,称为全局软件流水。与基本块软件流水方法相比,全局软件

流水方法要复杂得多,这也是目前研究得最多的一类软件流水问题。到目前为止,已经出现了数十种全局软件流水算法,归纳起来有如下三大类,分别是条件执行、条件组合和条件预测。对于条件执行方法,首先把条件分支上的操作转换成包含有条件的

操作,从而消去条件分支,成为基本块;然后在编译器中采用基本块软件流水方法进行调度;最后通过硬件的支持来执行那些带有条件的操作。条件执行方法最早出现于 Cydra-5 处理机中,目前已经在著名的通用处理机安腾 (Itanium) 和著名的嵌入式处理机 ARM 等中得到采用。条件组合要求在软件流水过程中对条件分支上的操作按照启动间距重新进行折叠和组合,并且生成新的条件转移路径。条件组合方法的优点是不需要硬件支持,其缺点是代码的膨胀很大,往往是指数级的。条件预测方法是选择转移概率比较高的一条路径作为主路径进行软件流水,它能够保证主路径上的指令级并行度很高,牺牲其他路径上的指令级并行度。

软件流水通常需要有适当的硬件支持,除了需要多个操作部件、多个指令译码器之外,还需要对循环控制、条件分支、寄存器换名和装入排空等给予支持。

总之,采用软件流水可以消除循环程序中的绝大部分数据相关,只需要保留占很小比例的强连通块数据相关,因此,指令级并行度很高,与循环展开等方法相比,其代码的膨胀也很小。软件流水方法已经在许多高性能编译器中得到广泛应用。

参考文献

Fisher J A and Rau B R. Instruct-level Parallel Processing. Science, 1991, 253: 1233 ~ 1241

(汤志忠)

ruanjian nixiang gongcheng

软件逆向工程 (software reverse engineering) 分析软件系统,确定其构成成分及各成分间的关系,提取并生成系统抽象和设计信息的工程。

软件逆向工程包含数据收集、知识组织和信息浏览三项规范活动。这三者构成软件逆向工程的基本过程。

数据收集 本活动中所收集的数据是指作为学习、推理和讨论基础的实际信息。原始数据是构作和浏览高层抽象的基础,因而数据收集是软件逆向工程的一项基本活动。

知识组织 本活动中的知识是指所知内容的总和,包括数据以及从数据中导出的关系和规则。所收集的数据必须按某种数据模型保存起来,以便实现有效的存储和检索,从而帮助分析人员实现对对象及其关系的分析。

信息浏览 遍历表征目标系统信息的多维空

间,按领域相关的标准分析和过滤信息,并以多种机制表达所得的信息,从而帮助分析人员进行程序理解过程中“假设—验证”的迭代过程。

软件逆向工程对软件测试、维护、鉴别、理解等诸多方面都有重要的辅助作用。(参见软件再工程)。

参考文献

1. Chikofsky Elliot J and Cross James H II. Reverse Engineering and Design Recovery: A taxonomy. IEEE Software, 1990, 7(1): 13 ~ 17

2. Tilley S R, Paul S and Smith D B. Towards a Framework for Program Understanding. In: Proceedings of the 4th Workshop on Program Comprehension (WPC '96: March 29 ~ 31, 1996; Berlin, Germany), 19 ~ 28. IEEE Computer Society Press (Order Number PR07283), March 1996

3. Scott R, Tilley. Coming Attractions in Program Understanding II: Highlights of 1997 and Opportunities in 1998. Technical Report, CMU/SEI-98-TR-001, Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, February 1998 (孙家骥)

ruanjian peizhi guanli

软件配置管理 (software configuration management) 一种按规则实施的管理软件开发过程和维护过程及其中间产品的方法。软件配置管理是软件生存周期中支持过程的一个关键组成部分,主要管理软件资源(包括与软件产品有关的各类文档、代码、数据、测试案例)及其演化,即管理软件生存周期中软件系统的构造和演化问题。

软件配置管理在整个软件生存周期中标识、定义系统中的软件配置项,系统化地控制对配置项的更改,记录和报告配置项的状态和修改申请,保证配置项的完整性、一致性和正确性,以及控制配置项的储存、装载和提交。其中,软件配置是指一个软件产品在软件生存周期各个阶段所产生的、机器可读或人工可读的各种形式和各种版本的文档、程序及其数据的综合。该综合中的每一个元素均称为该软件产品的软件配置中的一个配置项。

软件配置管理和软件版本控制是紧密相关的,版本控制是实施配置管理的基础。版本控制涉及了单个配置项的版本管理和软件配置的版本管理。

软件配置管理一般包括如下步骤:①标识 识别软件产品的结构、产品的构件及其类型,为其分配惟一的标识符,并以某种形式提供对它们的存取;

②控制 通过建立产品基线,控制在整个软件生存周期中对软件产品的修改和发布;③状态统计 记录并报告软件产品和修改请求的状态;④审计 确认产品的完整性,并维护一致性;⑤发布与提交 确保发布和提交产品的正确性及其发布信息。

一般而言,由于软件系统构造和演化的复杂性,有效的软件配置管理需要一套实用的管理规则,并结合相应的配置管理工具来实施。

参考文献

1. ISO/IEC 12207:1995 Information Technology-Software-Part1: Software Life-Cycle Process
2. Burrows C, George G, Dart S. Configuration Management. Ovum Ltd., 1996 (谢冰)

ruanjian sheji moshi

软件设计模式 (software design pattern)

对给定的应用环境中典型软件设计问题经过多次检验的解决方案的描述。大多数模式都只在确定的环境中工作,因此必须指定模式用于何处以及如何使用。设计模式这个术语是由 Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides 等四人于 1995 年提出的,用以捕获富有经验的设计者的设计解决方案,而这些设计解决方案已在许多项目中得到检验。使用模式可以复用其他开发者的成功经验,开发者能在以后的开发活动中反复应用模式。模式不只是局限于设计,也适用于包括系统分析、软件开发过程本身在内的许多领域。

一个好的模式通常应该做到以下几点:

- (1) 它通过解决方案解答问题;
- (2) 其解决方案经过多次实践的检验;
- (3) 其解决方案通常不是显而易见,而是需要洞察力;
- (4) 它描述多重结构中的(复杂)关系;
- (5) 它服务于一个有用的目的。

描述模式的元素一般可包含以下信息:

名字 捕获模式主要特性的描述性短语;

问题 模式适用的典型问题的本质;

应用环境 模式何时使用,如何使用(例如模式最适合应用的环境);

可应用性 模式的问题描述与现实环境匹配时的约束;

例子 实际使用模式来解决一些现实问题,有时附带实际的代码;

样板代码 在特定应用环境下模式实现的

例子;

基本原理 关于该解决方案为何适用于所描述的问题的简短解释;

相关模式 它们可以共享应用环境或约束,也可以联合使用。

描述设计模式有多种变形,但是上述列表包括了多数人认同的本质要素。

Erich Gamma 等四人总结了面向对象软件的设计经验,根据两条准则对面向对象软件的设计模式进行了分类。一条是目的准则,即模式是用来完成什么工作的。根据此准则,模式可分为创建型、结构型和行为型。创建型模式与对象的创建有关,结构型模式处理类或对象的组合,行为型模式描述类或对象如何交互以及如何分配职责。另一条是范围准则,即模式主要用于类还是对象。类模式处理类和子类之间的关系,这些关系通过继承建立,是静态的。对象模式处理对象之间的关系,这些关系在运行时可以改变,具有动态性。

参考文献

1. Marciniak John J. Encyclopedia of software engineering. John Wiley & Sons Inc., 2002
2. Gamma Erich, Helm Richard, Johnson Ralph, Vlissides John. Design Patterns: Elements of Reusable Object-Oriented Software. Addison Wesley Longman, Int., 1995 (钱乐秋)

ruanjian shengcun zhouqi

软件生存周期 (software life cycle) 软件产品或软件系统从产生、投入使用到被淘汰的全过程。

在计算机技术发展的初期,人们把软件开发简单地理解为编写程序。随着软件复杂性的增长,人们认识到软件开发活动应划分为需求分析、设计、实现、测试等若干个活动,并将这些活动以适当的方式分配到不同的阶段中去完成。不同的软件开发模型之间的区别在于将这些活动分配到不同的阶段中的方式不同。

早期,人们认识到一个软件系统生存周期也和人的一生相类似,可以划分为若干个互相区别而又彼此联系的阶段,上一阶段的结果为下一阶段的依据,上一阶段没有完成决不进行下一阶段的工作,从而形成了最早的软件生存周期概念。

通常把软件生存周期分为 5 个阶段,即需求、设计、实现(编码)、测试和维护。需求包括问题分析和需求分析,问题分析获取需求定义(又称需求规

约),需求分析生成功能规约;设计包括概要设计和详细设计,概要设计建立整个软件体系结构,包括子系统、模块以及相关层次的说明、每一模块的接口定义;详细设计产生程序员可用的模块说明,即数据结构说明及加工描述;实现是把设计结果转换为可执行的程序代码;测试包括单元测试、组装测试和确认测试,这些测试活动的目的就是使软件系统达到需求时提出的各项要求;维护是对投入运行的软件进行修改,使软件系统能适应外界环境的变化、实现功能扩充和质量改善。

随着对软件生产率和软件质量的要求不断提高,人们又认识到关键在于软件开发和维护中的管理问题,认为其关键是“软件过程”,为此进行了一系列的研究和讨论。IEEE 标准化委员会于 1991 年 9 月制定出“软件生存周期过程开发标准”,接着 ISO/IEC 于 1995 年制定出“信息技术——软件生存周期过程”标准(ISO/IEC 12207:1995)。

(朱三元)

ruanjian tixi jiegou

软件体系结构 (software architecture) 软件总体结构的抽象表示,或以此为研究对象的学科。软件体系结构具有如下几种含义。

规定性含义

软件体系结构由结构元集、结构形以及结构理三部分组成,即

软件体系结构 = (结构元集,结构形,结构理)

其中,结构元集为一组构成软件的结构元。结构元有三类,即处理元、信息元和连接元。处理元为对信息元施行处理的构件,信息元为处理元的处理对象,连接元负责构件间的连接。结构形包括特性、联系以及权重。特性用以约束结构元的选取,联系则约束结构元间的交互与组织,权重表示特性及联系的重要程度。结构理刻画体系结构人员选取体系结构风格、结构元、结构形的动因与根据。

体系结构风格是各种特定体系结构中结构元与结构形的抽象,它不如特定体系结构约束严格,亦不如特定体系结构完备。例如,有分布式风格,多进程风格等,它们强调的只是特定体系结构的某些方面。

描述性含义

软件体系结构由构件集、连件集、模式以及约束集四部分组成,即

软件体系结构 = (构件集,连件集,模式,约束集)

其中,构件集表示构成软件的一组组成元素,连

件集为一组连件,用以刻画各构件间的交互,模式为软件设计风格的描述,反映由构件及连件构成软件的构成原则,约束集中的约束表示对模式所加的限制条件。例如,在客户—服务器系统中,客户与服务器均为构件,构件间交互的描述(如过程调用、事件广播等)为连件,客户—服务器模式为模式,具体系统对模式所加条件为约束。

多视面含义

软件体系结构为软件的一个或多个结构,每一结构反映一种视面,即

软件体系结构 = 结构集

结构 = (构件集,外部可见特性集,联系集)

其中,构件集表示构成软件的一组组成元素,外部可见特性反映为其他构件可利用该构件所作的假定,联系用以沟通相关构件。

由于软件体系结构可有多个结构,从而可有多类构件、多种联系,故在定义中并不指明何类构件与何种联系。常用的结构类型有模块结构、进程结构和概念结构等。常用的视面有代码视面、模块视面、执行视面以及概念视面。其中惯常理解的软件体系结构反映了概念视面。

学科含义

以前述各种含义的软件体系结构为研究对象的学科或谓在研究与开发前述各种含义的软件体系结构中所涉及的理论、原则、方法、技术所形成的学科。

软件体系结构发展不久,迄今未见被普遍接受的单一定义,然而,它对软件的后续开发过程以及产品质量的影响举足轻重,已成为软件工程的重要研究方面,且其重要性将与日俱增。

参考文献

1. Perry D E and Wolf A L. Foundations for the Study of Software Architecture. In: ACM SIGSOFT Software Engineering Notes. 1992, 17(4)
2. Base Len etc.. Software Architecture in Practice. Addison-Wesley Publishing Company, 1997
3. Shaw Mary and Garlan David. Software Architecture: Perspective on an Emerging Discipline. Prentice Hall, 1996

(徐家福)

ruanjian wei hu

软件维护 (software maintenance) 软件产品在交付之后,为改正错误、改进性能或其他属性,或者为适应变化了的环境而对其进行修改的活动。近年来,人们更倾向于有些软件维护活动应在软件交

付前就开始。

在软件交付后的整个运行期间都可能发生软件的维护活动,所以,在整个**软件生存周期**中,软件维护阶段的时间通常要比软件开发阶段的时间长得多。同时,软件维护需要对现有的软件进行修改,而这种修改可能会影响到软件中未被修改的部分。因此,在整个软件生存周期的总成本中维护的代价是昂贵的。通常,维护的成本约占生存周期总成本的三分之二,而软件开发的成本约占三分之一。

软件维护大致可分为如下四类:

(1) 改正性维护——为改正发现的问题而修改软件的活动。

(2) 适应性维护——为使其在一个已变化的或将要变化的环境中保持可用而修改软件的活动。

(3) 完善性维护——为提高其性能或易维护性而修改软件的活动。

(4) 预防性维护——为在故障发生前检测并改正软件产品中隐藏的故障而修改软件的活动。

通常把适应性维护和完善性维护归为增强活动,把改正性维护和预防性维护归为纠正活动。增强性维护主要是由于需求和环境的变更而引起的,预防性维护以预防问题的发生为目的,它经常用于对安全来说是至关重要的软件产品。

在所有的维护活动中,改正性维护约占 20%,适应性维护约占 25%,完善性维护占 50% 以上,预防性维护不足 5%。由此可见,纠正性维护活动只占有所有维护活动的一小部分,大部分的维护活动都是增强性维护。

多年来已经出现过多种软件维护模型,这些模型大体上都包含三个阶段:理解软件、修改软件和重新验证软件。有些维护模型包括如下阶段:确定维护目标,理解软件,产生软件变更、说明波及效应以及进行回归测试。

ISO/IEC(1999a) 中软件维护主要包括如下活动。其中过程实现活动在软件交付前进行,其他活动在软件交付后进行:

(1) 过程实现活动 该活动制定维护过程中使用的计划和步骤,建立所需的配置管理组织的接口。

(2) 问题和修改分析活动 该活动评估修改请求以理解问题所在,提出解决方案,获得对实现指定解决方案的批准。

(3) 修改实现活动 该活动对软件产品实施修改并进行测试。

(4) 维护复审和验收活动 软件经修改后必须

进行复审和验收,以确保软件产品的正确性。

(5) 软件移植活动 当软件产品需要从旧的操作环境转移到新的操作环境上时,需进行软件移植活动。

(6) 软件退役活动 当已不再需要某软件,或者已开发出可取代该软件的新软件时,该软件必须按一种有序的方式解除它所提供的服务,实现该软件的退役。

参考文献

Marciniak John J. Encyclopedia of software engineering. John Wiley & Sons Inc., 2002 (钱乐秋)

ruanjian xitong

软件系统 (software systems) 计算机系统中由软件组成的系统。它包括操作系统、语言处理系统、数据库系统、分布式软件系统和人机交互系统等。操作系统用于管理计算机的资源和控制程序的运行。语言处理系统是用于处理软件语言等的软件,如编译程序。数据库系统是用于支持数据管理和存取的软件,它包括数据库、数据库管理系统等。数据库是常驻在计算机系统内的一组数据,它们之间的关系用数据模式来定义,并用**数据定义语言**来描述;数据库管理系统是使用户可以把数据作为抽象项进行存取、使用和修改的软件。分布式软件系统包括**分布式操作系统**、**分布式程序设计系统**、**分布式文件系统**、**分布式数据库系统**等。人机交互系统是提供用户与计算机系统之间按照一定的约定进行信息交互的软件系统。人机交互系统可为用户提供一个友善的人机界面。

发展过程

在第一台计算机于 1946 年出现后一段时间内,计算机没有任何软件系统,特别是没有操作系统。用户直接使用机器语言编制程序,并通过控制台开关系来调试和操作运行的程序。

20 世纪 50 年代后期起,计算机开始有较大发展,不仅速度显著提高,而且存储容量增长颇快,这就为软件的发展奠定了物质基础。在此期间,先后出现了 FORTRAN 和 ALGOL 60 等程序设计语言及其相应的编译程序,同时,大量出现了对计算机硬件和软件进行管理的软件——**管理程序**,例如,美国 IBM 360 系列计算机系统的初级控制程序和英国 1900 系列计算机的执行程序等。进入 70 年代以后,随着计算机应用的推广和数据处理的发展,有效的支撑数据共享的软件系统——数据库系统应运而生

生。关系数据库及其相关理论的研究,使得数据库开始实用化、商品化。同时操作系统、语言处理系统发展有了重大突破,如 UNIX 操作系统、FORTRAN、COBOL、C、PASCAL 等语言及其编译系统的广泛应用。

70 年代中期以来,计算机网络和分布式计算机系统发展较快。在分布式计算机系统研究、制造和付诸实用的过程中,分布式软件系统是首先被提出并着手研制的。分布式操作系统和分布式程序是构造分布式系统的基础之一,因而首先被人们关注,随后,为了发展分布式应用,分布式文件系统、分布式数据库系统等也相继提出。这样,分布式软件系统成了软件系统研制的热点。分布式软件系统的研究目标是:①如何有效、合理地分配、管理系统资源,特别是如何有效地利用分布式计算机系统主要的功能,如资源共享、并行处理等。②如何为应用提供良好的支撑环境(包括开发、调试和运行)。它和集中式软件系统的主要区别在于分布性和合作性。这几个问题在这 10 年中进行了充分的研究,其中,许多关键技术已得到解决。特别是在计算机网络的推广和应用中,出现了一批良好的网络软件,如:NETWARE, LAN MANAGER 等。

在软件系统快速发展和应用的基础上,特别是随着微型计算机、工作站以及网络的广泛应用,用户对软件系统的需求越来越迫切。因此,软件系统成为商品,开始规模化生产,形成产业,出现了一批崭露头角、擅长生产软件系统的著名企业,如以生产微型计算机操作系统而闻名的微软公司、以生产网络软件而闻名的 NOVELL 公司、以生产数据库系统而闻名的 ORACLE 公司、SYBASE 公司等。软件系统产业的出现和形成,为软件系统的发展注入了新的推动力。由于用户需求和市场推动,软件系统开始向开放化、标准化发展,如设计 UNIX 的 POSIX 标准,设计数据库的 SQL 标准,设计网络的 ISO 标准等(参见网络软件)。

基本内容

操作系统的功能包括处理器管理、存储管理、文件管理、设备管理和作业管理。其主要研究内容包括:操作系统的结构、进程(任务)调度、同步机制、死锁防止、内存分配、设备分配、并行机制、容错和恢复机制等。

语言处理系统的功能是各种软件语言的处理程序,它把用户用软件语言书写的各种源程序转换为可为计算机识别和运行的目标程序,从而获得预

期结果。其主要研究内容包括:语言的翻译技术和翻译程序的构造方法与工具,此外,它还涉及正文编辑技术、连接编辑技术和装入技术等。

数据库系统的主要功能包括数据库的定义和操纵,共享数据的并发控制,数据的安全和保密等。按数据定义模块划分,数据库系统可分为**关系数据库**、**层次数据库**和**网状数据库**。按控制方式划分,可分为(集中式)数据库系统、**分布式数据库系统**和**并行数据库系统**。数据库系统研究的主要内容包括:数据库设计,数据模式,数据定义和操纵语言,关系数据库理论,数据完整性和相容性,数据库恢复与容错、死锁控制和防止,数据安全性等。

分布式软件系统的功能是管理分布式计算机系统资源和控制分布式程序的运行,提供分布式程序设计语言和工具,提供分布式文件系统管理和分布式数据库管理系统等。分布式软件系统的主要研究内容包括分布式操作系统和网络操作系统、分布式程序设计、分布式文件系统和分布式数据库系统。

人机交互系统的主要功能是在人和计算机之间提供一个友善的人机接口。其主要研究内容包括人机交互原理、人机接口分析及规约、认知复杂性理论、数据输入、显示和检索接口、计算机控制接口等。

展 望

一台计算机硬件,在软件系统支撑下将形成一个用户的最基本平台——运行平台,不同的软件系统将形成不同功能,不同级别的运行平台,例如,在一台 PC 机上,装入 MS-DOS 操作系统就形成 DOS 平台,装入 UNIX 操作系统就形成 UNIX 平台,而根据不同的 UNIX 又可分别形成 SCO 平台,AT&T 平台等。所以,为了满足不同用户、不同应用的需求,软件系统的发展将趋向实用化、高效化、伸缩化、友善化、开放化和产业化等。(谢立)

ruanjian xuqiu dingyi

软件需求定义 (software requirements definition) 软件需求的完整陈述。它是软件开发人员与用户密切合作,了解用户的需要、目的和期望,并进一步表述而成的定义性陈述。有时又称为软件需求规约。它是用户与软件开发人员之间契约的基础,主要面向用户,采用基于现实世界的描述模型,以便于用户理解。

软件需求包括功能需求和非功能需求两个

方面。

功能需求从用户的角度明确软件系统必须具有的功能行为,其中包括系统的操作过程和操作模式的控制过程等描述。功能需求不仅要说明每项功能需要“做什么”,而且还要指明这些功能间的联系及相互的依赖关系(控制和数据),但不涉及“如何做”的描述,它是整个软件需求的核心所在。

在功能需求的基础上,非功能需求对软件需求作进一步刻画,包括功能限制、设计限制、环境描述、数据与通信规程和项目管理等。功能限制刻画软件系统的性能、响应时间、安全性标准和质量指标等;设计限制主要包括系统的开发平台等;环境描述主要包括系统所属环境的各个方面及其应用领域;数据与通信规程主要刻画系统各部分之间以及系统与外部环境之间的数据流;项目管理涉及系统开发与系统交付等方面的需求,主要包括文档标准、模块测试与集成过程、期限和可接收性标准等。

对上述软件需求所涉及的各个方面,其侧重点随软件类型而异。

好的软件需求定义文件应该是尽可能详细、完备、一致并具有较强的可适应性。其作用如下:

(1) 作为用户和软件开发人员建立合同的基础,它是双方对待解问题的共同理解;

(2) 作为软件开发的依据,开发人员据此写出相应的功能规约,然后再选择合适的解题途径,进行软件的设计与实现;

(3) 作为软件确认和验证的基础,一方面可据此确认系统是否满足用户需求;另一方面,可据此验证软件的设计与实现是否正确。

参考文献

1. McDermid J A. Software Engineer's Reference Book. Butterworth Heinemann Ltd., 1991

2. Partsch H A. Specification and Transformation of Program. Springer-Verlag, 1990 (董丽君)

ruanjian yuyan

软件语言 (software language) 用以书写计算机软件的语言。它主要包括需求定义语言、功能性语言、设计性语言、程序设计语言以及文档语言等。

需求定义语言用以书写**软件需求定义**,软件需求定义是软件功能需求和非功能需求的定义性描述。软件功能需求刻画软件“做什么”,软件非功能需求刻画诸如功能性限制、设计限制、环境描述、数

据与通信规程以及项目管理等。需求定义语言经历了从非形式的自然语言到半形式的语言及形式语言的发展,迄今半形式的需求定义语言已有较大进展,已逐步用于软件工程实践。

功能性语言用以书写**软件功能规约**,软件功能规约是软件功能的严格而完整的陈述。软件功能规约通常只刻画软件系统“做什么”的外部功能,而不涉及系统“如何做”的内部算法,因此,功能性语言通常又称为**功能规约语言**。从形式程度看,有非形式的功能性语言和形式的**功能语言**之分。前者是指未加限定的自然语言,后者则指其语法和语义均显式和精确定义的语言。从理论基础看,有代数类功能性语言和逻辑类功能性语言之分。前者指以异调代数、范畴论等代数理论为主要理论基础的规约语言,后者则指以一阶谓词演算等逻辑理论为主要理论基础的规约语言。功能语言涉及规约对象、规约方法以及规约性质等。规约对象主要包括过程抽象和数据抽象两类:过程抽象是指从输入值集到输出值集的映射,其定义域和值域均由数据抽象刻画。数据抽象则提供了数据值集及其上的运算符集。规约方法涉及如何对过程抽象与数据抽象进行规约。目前,功能性语言的研究进展较大,理论基础日趋成熟,已逐步用于软件工程实践。

设计性语言用以书写**软件设计规约**。软件设计规约是软件设计的严格而完整的陈述。一方面,它是软件功能规约的算法性的细化,刻画了软件“如何做”的内部算法;另一方面,它又是软件实现的依据。从细化程度来看,有总体设计规约与详细设计规约之分。前者刻画设计的总体架构;后者刻画详尽实现细节。这两种设计规约一般都是用形式体系刻画的,亦即,都是形式的。设计性语言的发展已相对成熟,并已用于软件工程实践。

实现性语言,即一般的**程序设计语言**,用以书写计算机程序,而计算机程序是计算任务的处理对象和处理规则的描述。任何以计算机为处理工具的任务都是计算任务。处理对象是数据或信息,处理规则反映处理动作和步骤。程序设计语言的基本成分是:①数据成分,②运算成分,③控制成分,④传输成分。程序设计语言有多种分类法,例如,按语言级别分,有**低级语言**与**高级语言**,按应用范围分,有通用语言与专用语言。按成分性质分,有顺序语言、并发语言、并行语言、分布语言。按使用方式分,有交互式语言与非交互式语言等。程序设计语言的发展已有近 50 多年的历史,已相当成熟。

文档语言用以书写软件文档。前述的软件需求定义、软件功能规约、软件设计规约等都是软件文档。此外,还可能有一些其他的阐明性资料,以便于读者理解相应软件,它们都是软件文档。这些阐明性资料一般都是用自然语言或者半形式的语言书写的。

(徐家福)

ruanjian zai gongcheng

软件再工程 (software reengineering) 通过软件逆向工程、重构和正向工程,将现有的软件系统重新构建,以适应新的需求的工程。广义上说,任何可以改进人们对软件的理解和改进软件本身的活动都是软件再工程的内容,如图1所示。

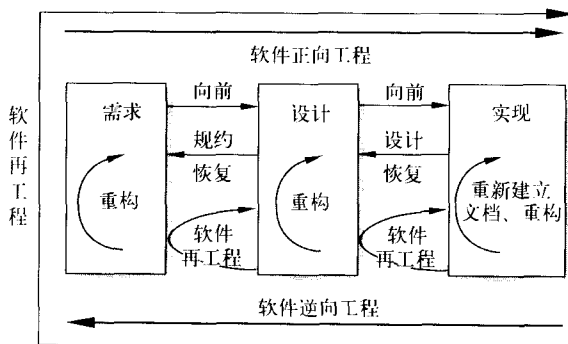


图1 软件再工程定义

由于软件再工程复用了已有的软件资源,因而往往可以以更少的开销、更短的时间、更低的风险把软件系统改造成一个新的系统,从而在操作、系统能力、功能、性能或易维护性和可支持性上得到改进。

软件再工程的基本框架如图2所示,它给出了进行一次软件再工程实践的全过程中所应考虑的活动。但这样的框架所给出的并不是一个需要严格遵守的活动规范,针对不同的具体系统,可能需要进行相应的修改才能适应具体问题的需求。

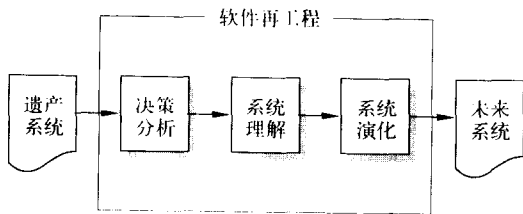


图2 软件再工程的基本框架

决策分析 决策分析的活动包括:对原有系统的详细审查和分析,明确对于改造后的新系统的需求,确定软件再工程活动的范围和方向等。决策的重要目的是在软件再工程、维护和新开发之间进行选择,最终决定是否要进行软件再工程。

系统理解 如果决策分析的结论是应该进行软件再工程,就要开始对原有的系统进行系统理解。目前,所谓的遗产系统积累得越来越多,这类系统的大部分文档和设计信息都由于时间久远而丢失,因此,我们需要通过对原有系统的源代码、设计记录以及其他文档资源的分析,才能得到原有系统的全面而详细的信息,为下一步的转化工作提供坚实基础。

系统演化 通过改造和重组,将原有系统转化为新的系统。对于一个大的系统,针对其不同部分和所期望的新系统之间的距离,可以使用不同的演化策略,主要有维护、改造和替换三种方法。

参考文献

1. Arnold R S. Software Reengineering. IEEE Computer Society Press, 1993

2. Feiler P H. Reengineering: An Engineering problem (CMU/SEI-93-SR-5, ADA 267117). Pittsburgh, pa: Software Engineering Center, CMU, July 1993

(孙家骥)

ruanjian zhiliang

软件质量 (software quality) 反映软件系统或软件产品满足明确或隐含需求能力的特性的总和。其含义有四:其一,能满足给定需要的特性的全体;其二,具有所期望的各种属性的组合的程度;

其三,顾客或用户觉得能满足其综合期望的程度;其四,软件的组合特性,它确定软件在使用中将满足顾客预期要求的程度。简言之,软件质量是软件一些特性的组合,它仅依赖软件的本身。

对于软件质量有三种不同的视面,用户主要感兴趣的是如何使用软件,软件性能和使用软件的效果。所以他们关心的是:①是否具有所需要的功能;②可靠程度如何;③效率如何;④使用是否方便;⑤移植到另一环境是否容易。开发者负责生产出满足质量要求的软件,所以他们对中间产品的质量以及最终产品质量都感兴趣,当然开发者视面还必须体现软件维护者所需要的质量特性。对于管理者也许要注重总的质量而不是某一特性,他还须从管理准则,如在进度拖延或成本超支与质量的提高之间进行权衡,以达到用有限的成本、人力和时间使质量达到优化的目的。

软件质量可以用下列特性来评价:

(1) 功能性 与一组功能及其指定的性质有关的一组属性。这里的功能是指满足明确或隐含的需求的那些功能。

(2) 可靠性 在规定的一段时间和条件下,与软件维持其性能水平的能力有关的一组属性。

(3) 易使用性 与为使用软件所需作的努力和由规定或潜在的用户对这样的使用所作的个别评价有关的一组属性。

(4) 效率 在规定的条件下,与软件性能水平和所使用资源量之间的关系有关的一组属性。

(5) 易维护性 与进行指定的修改所需的努力有关的一组属性。

(6) 易移植性 与软件可从某一环境转移到另一环境的能力有关的一组属性。

软件质量度量 能被用来确定软件系统或软件产品某特性值的一种定量尺度。最初由 Rubey 和 Hartwick 于 1968 年提出一些属性的度量方法,但未建立质量度量模型。Boehm 等人于 1976 年提出了定量地评价软件质量的概念,提出了 60 个质量度量公式,并首次提出了软件质量度量的层次模型。1978 年 Walters 和 McCall 提出了从软件质量要素、准则到度量的三层次式的软件质量度量模型,将质量要素降到 11 个。国际标准化组织于 1985 年建议的质量度量模型是:高层——软件质量需求评价准则;中层——软件质量设计评价准则;低层——软件质量度量评价准则的三层次模型。高层由 8 个元素组成。上海计算机软件技术开发中心于 1988 年提

出了软件质量评价体系,即如前述的 6 个软件质量特性,选用 22 个评价准则,以及每一度量由若干度量元组成的度量族,并给出了评价准则与质量特性的关系和应用策略,国际标准化组织(ISO)和国际电工委员会(IEC)于 1991 年提出了标准,给出了 6 个特性和 21 个子特性。

软件质量评价过程 用前述软件质量特性来评价软件质量的主要步骤:一般由质量需求定义、评价准备和评价过程三步组成。质量需求定义表达了环境对被评价软件的要求,它必须在开发前就被定义。评价准备包括质量度量的选择,评价准则的选择等等,以及评定等级的定义和评估准则定义。评价过程一般可再细化为三步,即测量、评级和评估。

参考文献

1. 朱三元等编. 软件质量及其评价技术. 北京:清华大学出版社,1990
2. ISO/IEC 9126 — 1991 Information Technology — Software Product Evaluation — Quality Characteristics and Guidelines for Their Use (朱三元)

ruanjian zhongjianjian

软件中间件 (software middleware) 建立在具有基本通信协议的操作系统之上,支持应用程序的有效开发、部署、运行和管理的支撑软件。

在网络应用需求的带动下,软件中间件于 20 世纪 80 年代中后期形成,并在 90 年代得到蓬勃发展。80 年代,基于 TCP/IP 协议的进程间通信机制及其称为 Socket API 的应用程序设计接口是人们在网络上开发分布式应用的主要手段。与 60 年代前在裸机上用汇编语言编写程序一样,用 Socket API 开发分布式应用十分繁琐、复杂和低效。人们开始研究便于分布式应用开发、部署、运行和管理的软件中间件。早期的软件中间件以支持局域网环境中的资源共享(特别是数据共享)为主要目的;进入 90 年代,以支持异构环境中的应用互操作为主要目的的软件中间件得以发展;世纪之交,面向应用领域的软件中间件成为被关注的焦点。

软件中间件一般包括:①便于分布式应用实体之间交互的高级通信机制;②支持分布式应用有效运行的共性基础服务,例如支持定位的名字服务、支持分布式事务管理的事务处理服务、支持分布式时间管理的时间服务等;③开发、部署和管理分布式应用的基本工具,例如接口定义与编译工具、应用部署工具等。

软件中间件种类繁多。从通信机制和程序设计风范的角度看,软件中间件可以分为面向消息中间件、远程过程调用中间件和面向对象中间件。

(1) 面向消息中间件 支持分布式应用实体之间采用消息传递的异步方式进行通信,即消息发送者把消息提交给中间件之后,无需等待消息接受者的应答。面向消息中间件是基于包交换的底层通信模式的自然扩展。面向消息的中间件一般提供队列管理、优先级管理、路由管理、存储转发、可靠性管理以及信息的发布与订阅等服务。面向消息中间件适合于松耦合、大范围的信息交互应用。IBM 的 MQ-series 是 90 年代典型的面向消息中间件。

(2) 远程过程调用(RPC)中间件 中间件支持客户-服务器计算模式的分布式应用,其中服务器应用以过程的形式定义,客户采用 RPC 机制请求异地服务器上的过程。RPC 是一种请求-响应的同步通信机制,客户(即请求方)必须等到服务方的响应结果后才能处理后续业务。DCE(分布计算环境)中的 RPC 是典型的远程过程调用中间件。远程过程调用中间件是过程式程序设计风范在分布式应用中的自然扩展。由于这种程序设计风范在模块化等方面的固有局限性,远程过程调用中间件逐步被面向对象中间件所替代。

(3) 面向对象中间件 支持面向对象的多层客户-服务器模式的分布式应用,其中服务器应用以对象的形式定义,每个对象用接口定义语言(IDL)定义明晰的访问接口,客户通过对象请求代理(ORB)访问异地服务器上的对象。服务器中的对象不仅能够被访问,而且自身也可能作为其他服务器中对象的客户。因此在面向对象中间件中,客户与服务器的角色划分是相对的或多层次的。ORB 是远程过程调用(RPC)机制在面向对象模型中的自然扩展。与 RPC 类似,ORB 是一种请求响应的同步通信机制。在面向对象中间件中,共性服务按照面向对象的方式定义。面向对象中间件提供了构件化的开发和部署工具。OMG 公司的 CORBA、SUN 公司的 Java/RMI 以及 Microsoft 公司的 DCOM 是 90 年代面向对象中间件的三个有代表性的技术体系。

随着软件中间件的广泛应用,其内涵和外延不断拓展。从应用领域的角度看,软件中间件又可分为通信中间件、应用服务器与构件管理平台、数据集成中间件、应用集成中间件以及流程管理中间件等。其中通信中间件包括了面向消息中间件、远程过程调用中间件和面向对象中间件。

软件中间件仍处在十分活跃的发展时期。支持组织之间通过 Internet 实现数据、应用和流程集成的软件中间件将得到进一步的发展,其中值得重视的技术是 Web Service 技术和 Grid 技术;为了适应更加复杂多变的应用环境,具有动态适应能力的软件中间件将进一步发展,其中主体技术更应重视;支持复杂分布式应用开发的工具和环境将在软件中间件中占据更重要的位置,其中值得关注的技术是 OMG 公司的模型驱动的体系结构(MDA)。

参考文献

Kurt Geihs. Middleware Challenges Ahead. Computer, June 2001: 24 ~ 31

(王怀民)

ruanjian zhuti

软件主体 (software agent) 封装的软件实体。其基本特性如下:

自主性 能在无外界直接干预的情况下独立运行;

反应性 能主动而有选择地观察环境,及时采取行动,适应环境变化;

(本原)主动性 具有表现目标制导行为的能力,必要时采取主动;

交往性 具有和其他软件主体交往之能力,一般较为主动。

软件主体可在开放、动态和多变的环境下完成指定的任务。其基本工作方式是获取环境中的信息或感知环境的动态变化,选取合适的行为方式,与其他软件主体协同与合作,对外部环境加以反应和作用,相应对内部有关机制适应调整。

面向对象技术起过且正在起着重要作用,但自 20 世纪 80 年代以来,从开放、动态和多变环境的角度考察,面向对象技术呈现出某些不足,主要表现在:自主性局限、反应性被动、封装性固定等。从而,出现了软件主体技术。对象和软件主体相比较,对象的自主性较为局限,在接收外界发送的消息后,即采取相应的动作,软件主体则对外界请求执行与否,自身有决定权,故其自主性较强;对象的封装性固定,其边界对环境全开放,软件主体只部分开放;对象对环境之反应被动,软件主体则能主动观察环境变化,做出反应。此外,对象缺乏目标制导行为之能力,软件主体则有这种能力。

近年来,随着万维网平台的快速发展,人们迫切需要新的软件基础技术来应对万维网平台的开放、动态和多变的特征,为软件开发平台、运行平台和开

放式网络应用提供有效的基础技术支持,从而使软件主体技术成为软件新技术研究的主流之一。很多政府机构、民间团体、大学研究机构和公司参与了软件主体技术的研究,著名的有 FIPA, UMG, CLB-FATE, USDARPA 和 EUAgentlink, CMU, MIT, Microsoft, IBM 等。

参考文献

Wool dridge Michael and Cianearini Paolo. Agent-oriented software engineering; The state of the art. PZiancarkli and Wooldridge M., editors, Agent-Oriented software Engineering, Lecture Notes in Artificial Intelligence. Springer Verlag, 2001, 1957:1~28 (吕建)

ruanjian zidonghua fangfa

软件自动化方法 (software automation method) 尽可能借助计算机系统实现软件开发的方法。计算机系统除泛指一般系统外,尤指用于软件开发的系统,特别是软件自动化系统。“尽可能”一词反映软件开发的自动化程度。软件开发是指除维护阶段外的软件生存全期,即从非形式的软件需求定义,经形式的软件功能规约、软件设计规约到可执行的程序代码、调试及至确认、交付使用的全过程。

软件自动化也可狭义地理解为,从形式的软件功能规约到可执行的程序代码这一过程的自动化。可执行的程序代码既可指低级语言程序代码,也可指高级语言程序代码。此外,自动化的程度是相对的,其高低一般因系统而异。

软件自动化一词几乎和软件一词同时出现,原称自动程序设计。早在 20 世纪 50 年代,程序人员从程序设计实践中深感程序设计工作的繁琐、不易、低效,便试图在可能范围内将一些机械性工作委交机器本身去做。在那时,实现高级语言的编译就是自动程序设计,如 1956 年美国国际商业机器公司 (IBM) 建立的第一个实用的 FORTRAN 编译程序曾被称为自动程序设计系统。60 年代,出现了编译程序的编译程序和各种自编译程序。软件工程出现后,软件自动化的含义得到较大发展,其自动化的内容涉及到软件生存全期的各个阶段。

软件自动化分三个不同层次:一为低级自动化:自动化系统只起程序人员的作用,亦即,从软件设计规约到可执行的程序代码这一过程的自动化。二为

中级自动化:自动化系统除了起程序人员的作用外,还起设计人员的作用,甚至起部分系统分析人员的作用。亦即,从形式的软件功能规约、到设计规约、直到可执行的程序代码这一过程的自动化。三为高级自动化:自动化系统除了起程序人员、软件设计人员、系统分析人员的作用外,还起部分领域专家的作用。亦即,从非形式的软件需求定义,经形式的软件功能规约、软件设计规约,直到可执行的程序代码这一全过程的自动化。

基本内容

主要涉及软件开发、软件规约、自动生成和自动验证等。

软件开发 从软件的需求定义,经形式的软件功能规约、设计规约、到可执行的程序代码,及至确认、交付使用的全过程。亦即,维护阶段除外的软件生存全期。它涉及开发风范和开发模型。开发风范指的是,软件开发的准则与风格。如,自顶向下的功能分解风范,自底向上的面向对象风范等。开发模型指的是,软件开发风范的结构体现。如,功能分解风范中的瀑布模型,面向对象风范中的喷泉模型等。开发模型由开发方法和工具来体现。方法和工具是一个问题的两个方面。方法有全局性方法和局部性方法两类。工具有需求分析工具、设计工具、实现工具、测试验证工具等多种。

软件规约 软件所应满足的要求的陈述。它是软件开发的依据,也是软件自动化的依据。根据陈述的性质与层次,又可区分为需求规约(即需求定义)、功能规约和设计规约等。它们分别用相应的软件语言来描述。目前,需求规约尚难以完全用形式体系刻画,一般借助自然语言。功能规约与设计规约可分别采用形式的功能性语言和设计性语言来刻画,习惯上称之为形式功能规约和形式设计规约。软件自动化要求软件规约和软件语言应有利于机器的自动处理,易于实现从软件规约自动或半自动地导出正确的程序。

自动生成 从需求规约或功能规约(取决于软件自动化的不同层次),由相应的软件自动化系统“自动”生成可执行的程序代码。这项工作的难度颇大,迄今由需求规约过渡到功能规约,还难以自动生成,尚须借助“人与系统”的交互。由功能规约到设计规约,原则上可以自动进行,而实际上,仍需借助“交互”。至于由设计规约(指详细设计规约,而

不是概要设计规约)到可执行的程序代码则可自动进行,而且技术已相对成熟。软件自动生成的难度与软件自动化的程度有关,软件自动化系统的自动化程度越高,难度越大。另一方面,针对某一应用领域而设计的专用软件自动化系统实现较易;相反,通用软件自动系统则实现较难。自动生成是软件自动化系统的核心,也是困难的关键所在。

自动验证 软件的正确性是相对其功能规约而言的。凡是具备且仅具备相应功能规约中所列出的功能的软件就是正确的软件。自动验证包括三方面的工作:第一,功能规约的正确性的自动验证。功能规约的正确性是针对需求规约而言的,但由于目前需求规约还难以完全用形式体系刻画,其自动验证尚处于探索阶段。第二,可执行的程序代码的正确性的自动验证。第三,从功能规约到可执行的程序代码转换过程的正确性的自动验证。理论上说,在肯定功能规约正确的前提下,只需验证“转换过程”的正确性,作为其转换出的结果,可执行的程序代码也就必然正确。自动验证是软件自动化系统必不可少的内容,但真正实用的验证方法仍不多见。

实现方法

主要有演绎综合、程序转换、归纳综合和过程实现等方法。

演绎综合 从给定的软件规约借助演绎推理综合出程序的方法。它将数学中的构造性证明与软件开发相联系,将开发步骤解释为证明步骤,从证明中抽取相应的程序。

代表性的工作是 Z. Manna 和 R. Waldinger 的 DEDALUS 演绎算法综合系统,它从用数理逻辑形式刻画的程序输入输出规约,即前置断言和后置断言,用演绎的方法自动导出等价的 LISP 程序。

程序转换 由一程序转换至满足其功能要求的另一程序的方法。程序转换有两类:一类是纵向转换,即由一抽象级别较高的程序转至另一满足其功能要求的抽象级别较低的程序。另一类是横向转换,即在相同(或类似)抽象级别上程序间的转换。

纵向转换的代表性工作 70 年代中期西德慕尼黑技术大学信息学研究所 F. L. Bauer 教授主持下开始研究的软件自动化系统,即计算机辅助、直觉指导的程序设计(CIP)项目。CIP 的目标是开发可形式保证程序正确性的程序开发系统。

横向转换的代表性工作 是英国爱丁堡大学 R. M. Burstall 和 J. Darlington 等人研制的 POP-2 系统和在此基础上发展起来的 ZAP 系统。这类系统根据一组规则基本自动地将作用式递归程序转换成高效的循环程序。

归纳综合 借助反映程序性态的实例,利用归纳推理导出程序的方法。其方式主要有两种:一种是“输入输出对”法:借助给出的一组输入输出对,逐步推导出适于一类问题的程序。另一种是“部分轨迹”法:通过所给实例的运行轨迹,逐步导出程序。

过程实现 借助过程化手段将一软件规约转化成目标代码的方法。在对应软件规约中的各个成分,其转换目标的相应成分明确,而且相应的转换映射也明确的前提下,该映射可借助过程来实现。

现状与展望

三十多年来,虽然软件自动化的含义不断深化,工作不断提高,但受目前计算机科学技术及有关学科发展的限制,现状还远远不能令人满意。现有多数系统均为实验性的,很少臻于实用,通用软件自动化系统更是如此。其状况表现为:第一,虽然有关软件开发风范与模型的工作不少,但从软件自动化角度,何种开发风范与模型更为合适,却未见系统性的研究工作。第二,软件规约语言的研究工作众多,包括设计性规约语言,功能性规约语言,广谱性规约语言。而需求性规约语言很少,且多半带有自然语言成分。第三,自动生成是软件自动化的核心与关键。问题是:①从功能规约生成相应的设计规约还难以达到完全自动化,尚需借助人系统的交互;②从需求规约生成相应的功能规约的多数系统基本上仍然停留在手工阶段。第四,自动验证是软件自动化必要内容。迄今,理论上虽然可以验证生成过程的正确性,但在实际软件开发过程中却很难验证。

为使软件自动化系统达到便于使用、功能较强和功效较高等目标,需进一步开展的研究有:①语言,特别是自然语言。由于人与系统的接口宜用自然语言描述,故必须研究自然语言理解与处理。为此,一方面,要研究自然语言的形式化;另一方面,也要研究形式语言的自然化。否则,不仅软件自动化的推广应用难以实现,而且社会信息化也将流于空谈。②人工智能,特别是机器学习。研究机器学习,如果不将机器学习的原理与技术融于软件自动化的

研究中,就不可能使系统能具备自适应、自学习以及自组织的能力。③数理逻辑,特别是动态逻辑。古典数理逻辑宜于描述静态系统,但是,为了描述软件生成,必须研究动态逻辑,借以奠定软件自动化的理论基础。

软件自动化是解决软件功能不强,质量欠佳和生产率低三大问题的新技术,是提高软件生产率的根本途径。随着计算机科学技术及有关学科的发展

和软件自动化技术的自身完善,它在未来的软件开发过程中必将起重要作用。

参考文献

1. 徐家福,陈道蓄,吕建,王志坚. 软件自动化. 北京:清华大学出版社,1994
2. Balzer R A. 15 Year Perspective on Automatic Programming. IEEE Software Engineering, 1985, 11 (11):1257 ~ 1268 (徐家福 王志坚 张家重)

S

sanwei donghua

三维动画 (3-D animation) 生成三维空间中场景及形体随时间而演变的一系列可供实时播放的画面的过程和技术。三维动画是计算机动画的一个主要分支。在三维动画中,三维模型由计算机构造,并通过对模型、虚拟摄像机、虚拟光源和物体材料的运动控制,由真实感图形绘制算法自动生成一系列逼真的连续动态图像,故亦称为模型动画。

在三维计算机动画中,三维场景主要由3种类型的实体组成:物体、摄像机、光源。每种物体都有一些特征,这些特征可按照一定的规律随时间而演变。使用这些特征可模拟物体的运动、变化等。例如,对物体而言,其特征有位置(位置随时间的改变可模拟物体的平移)、方位(方位的变化可模拟机器人手臂的旋转运动)、大小(大小的变化可模拟植物生长的演变等)、形状(形状的变化可模拟云彩的变化、心脏的运动、物体的变形等)、颜色(颜色的变化可模拟火焰、日出等)、透明度(模拟雾的效果等),等等;对于摄像机,其特征有视点位置、摄像机朝向等;对光源,其特征有光源位置、强度、衰减系数、类型(点光源、线光源、面光源或体光源)等;物体材料属性包括漫反射系数、镜面反射系数、自发光系数、会聚指数、透明度、反射系数、折射率、各种纹理等。

三维动画主要研究物体的运动控制技术以及与三维动画有关的造型、绘制等技术。三维动画技术大致可分为**关键帧动画**、**渐变和变形体动画**、**过程动画**、**关节动画**和**人体动画**、**基于物理的动画**等。

参考文献

1. 鲍虎军,金小刚,彭群生等. 计算机动画的算法基础. 杭州:浙江大学出版社,2000
2. Watt A, Watt M. Advanced Animation and Rendering Techniques: Theory and Practice. London: Addison-Wesley, 1992 (王裕国 金小刚)

sanwei kongjian shujuchang keshihua

三维空间数据场可视化 (visualization of data sets in 3D space) 用计算机图形学及图像处

理技术在计算机屏幕上显示三维空间数据场的图像的过程和技术。科学计算、工程计算及测量结果往往以三维空间中离散的数据场来表示,将数据场的整体或感兴趣的部分以图像形式在屏幕上显示出来是实现**科学计算可视化**的核心问题。

三维空间数据就其性质而言,可以分为标量场和向量场;就其相互关系而言,可以分为结构化和非结构化两大类。前者指数据场中的数据在逻辑上是以三维矩阵的形式组织起来的,而后者却无这一限制,甚至可以是散乱的。结构化的三维空间数据场还可以细分为规则数据场和不规则数据场等。

三维标量场的可视化技术已基本成熟,尽管其算法与数据场的类型有关,但其基本原理却是一致的。可以分为两类:第一类算法首先由三维空间数据场作出中间几何图元(如平面、曲线、曲面等),然后再由传统的计算机图形学技术实现画面绘制。这类算法可以利用现有的由硬件实现的画面绘制功能。图形生成及变换速度较快,且可以产生比较清晰的等值面(参见**等值面构作技术**)图像。第二类算法并不构作中间几何图元,而是由离散的三维数据场直接产生屏幕上的二维图像,称为**体绘制技术**。体绘制技术的实质是重新采样及图像合成,它的优点是可以产生三维数据场的整体图像,而不是仅仅显示出人们感兴趣的等值面。同时,也便于进行并行处理。人们往往根据不同的数据类型及显示要求选择不同的显示算法。

三维向量数据不仅有大小,而且有方向。如何在屏幕上形象、直观、快速地显示出三维向量场,仍然是科学计算可视化中的一个研究热点。已经提出的是基于几何形状、基于颜色、基于纹理的向量场可视化算法。

参考文献

1. Nielson G M, Rosenblum L J. Visualization in Scientific Computing. IEEE Computer Society Press, 1990
2. 石教英,蔡文立. 科学计算可视化算法与系统. 北京:科学出版社,1996
3. 唐泽圣. 三维数据场可视化. 北京:清华大

学出版社,1999

(唐泽圣)

sanwei wangge chuli

三维网格处理 (3D mesh processing) 处理多边形网格特别是三角形网格模型的技术。它是数字几何处理(DGP)中的重要内容。随着三维几何数据的获取技术逐渐成熟,研究的焦点主要集中在狭义的 DGP 技术,即基于网格的几何信号处理框架。现有的算法框架主要有 3 大类:①基于半规整或者称为具有细分拓扑连接性的网格方法;②基于松弛算子的方法;③拓扑映射的方法。

基于半规整网格的方法是以小波理论和多分辨率分析为基础的,如细分小波和球面小波。其基本思想是把一个复杂的函数分解为一个简单的低分辨率部分与若干相继的小波系数部分的叠加。因从三维重建算法和各种造型软件得到的网格模型通常不具备细分连接性。为此研究者提出了一系列对网格重新网格化的方法,其主要步骤包括数据分片、参数化和重采样。目前已经实现的基于细分网格技术的典型 DGP 技术应用包括多分辨率编辑、几何压缩和网格变形等。

基于松弛算子的方法代表性工作是基于信号处理的光顺造型算法和基于非均匀松弛算子多分辨率信号处理框架。前者从网格的局部邻接信息中导出了一种几何信号的离散傅里叶变换,并在其相应频域中设计带通滤波器或称为松弛算子,达到网格光顺的目的。后者的贡献在于设计了一种依赖于几何和拓扑连接的非均匀松弛算子。利用该算子,结合渐进网格的多分辨率表示,可以设计出适用于任意网格的多分辨率变换,用于网格的光顺、增强、编辑和纹理映射。

拓扑映射方法先将网格变换到简单的定义域中,然后在新的定义域中实施信号分析。如几何图像方法,将网格自动剖切为开网格,然后参数化到矩形区域上,所有网格的信号,如顶点位置、法向和颜色等,都可以均匀采样成图像。类似地,可以将网格映射到球面上,利用球面调和函数,得到完整的分析和处理效果。

三维网格处理的另一个重要内容是对网格的编码和压缩。由于应用的不同,研究者除了采用以上方法进行编码和压缩方法外,也针对顶点间拓扑连接考虑合理编码,如入度编码技术等。某些编码可形成多边形条带,以利于图形硬件进行快速绘制。

参考文献

1. Lee A, Sweldens W, Schröder P, Cowsar L and Dobkin D. MAPS: Multiresolution Adaptive Parameterization of Surfaces. In: Proceedings of ACM SIGGRAPH 1998, 95 ~ 104
2. Taubin G. A Signal Processing Approach to Fair Surface Design. In: Proceedings of ACM SIGGRAPH 1995, 351 ~ 358
3. Gu X F, Gortler S J and Hoppe H. Geometry Images. In: Proceedings of ACM SIGGRAPH 2002, 355 ~ 361

(鲍虎军 张宏鑫)

sanwei wangge moxing canshuhua

三维网格模型参数化 (parameterization for 3D mesh model) 构造从三维曲面一部分到二维欧氏空间区域映射的过程和技术。它是几何处理过程的重要一环。网格模型参数化对如何用一张曲面逼近另一张机械制造中特定曲面的数值仿真、曲面的重采样和编辑、曲面的纹理映射等都有非常重要的意义。

多数参数化方法通过某种能量函数的极小定义来构造出具体的映射函数。这种能量可以是度量如何降低变形扭曲,保持角度或者保持面积,也可以是弹性能量。为使得所定义的能量极小,通常可以通过确定边界和交互约束,将约束优化问题转化为求解以投影坐标为未知量的大型线性方程组。幸运的是,所得到的大型矩阵一般较稀疏,利于快速数值求解。也有研究者通过两步法,在映射函数之后再添加一个变形函数,以减少参数化的整体变形问题。

同时,参数化可看成降维问题的一个特例。如有些研究者通过求解曲面上顶点间的测地距离,然后利用经典 MDS 将三维网格降维映射到二维平面,得到一种自由边界的参数化方法。另外,也有研究者直接采用构造映射的方法实现网格的参数化。

当然,由于参数化的目的在于构造从复杂曲面到简单定义域的映射,以利于分析和存取,因此其目标空间并不只限于二维平面,如近来出现的球面参数化技术。另外,受限于拓扑结构,当输入网格的拓扑较为复杂时,往往无法对模型直接作参数化处理。合理的做法是使用各类分片技术,然后再对各子面片进行参数化,如在基于小波分析的几何处理框架中,原网格就被参数化到简化后的基网格上。

参考文献

1. Burak Aksoylu, Andrei Khodakovsky and Peter Schröder. Multilevel Solvers for Unstructured Surface Meshes. In review, SIAM J. Sci. Comput

2. Zigelman G, Kimmel R, Kiryati N. Texture Mapping Using Surface Flattening via Multidimensional Scaling. IEEE Trans. Visual. Comput. Graphics, 2002, 4 ~ 6

3. Khodakovsky A, Litke N and Schröder P. Globally Smooth Parameterizations with Low Distortion. ACM Transactions on Graphics, 2003, 22

(鲍虎军 张宏鑫)

sanwei wangge moxing jianhua

三维网格模型简化 (simplification for 3D mesh model) 对给定网格构造相对粗略的逼近版本, 使能很好地保持原有网格几何特性的技术和算法。它是网格处理的重要手段之一。由于激光扫描仪等获取的网格数据, 往往数据规模庞大, 超过了现有计算系统的处理能力。为使得网格数据得到有效的存储、传输和计算, 网格简化成为重要研究内容。除了作为网络传输, 存储和显示用途外, 网格简化也是许多网格参数化方法的预处理步骤。由于三角形网格简单易用, 是数字几何处理中最广泛采用的数据形式, 也是三维网格模型简化的主要对象。

三维网格模型简化通常按某种规则, 逐步地合并和删除一类拓扑元素来达到数据简化的目的。按操作的中心对象可划分为基于顶点删除、边塌缩、面删除和顶点归一化等几类简化方法。基于顶点删除的简化方法, 每次缩减一个顶点及其相关的边和面, 然后使用 Delaunay 三角化方法来填补顶点删除后所形成的空洞。类似地, 也可以以边为中心, 实现简化。边塌缩算法则只需合并当前的边所连接的顶点和删除两个退化的三角形; 半边塌缩算法只将边连接的一个顶点作为规则判别对象, 而将另一顶点塌缩吸收。面删除算法中较典型的是三角形删除法和面合并算法。比较特殊的是顶点归一化方法, 它并不要求待合并的两点有边相连。

简化过程中的误差控制规则非常重要, 它可以用于选择更好的删除元素和简化效果, 度量简化过程的质量, 和给出简化的终判条件。目前已有许多很好的简化方法, 如基于二次误差度量的简化方法, 以边塌缩和顶点分裂为互逆操作的渐进网格框架、

通过延法线方向度量向内和向外的偏移量的包络法, 以及各种保持体积或者表面积、曲率、特征边的个数和入度等度量的简化方法。另外, 除了基于几何信息的简化外, 许多研究者还考虑了基于表面纹理和颜色的简化方法。总之, 简化过程的关键在于选取合适简化的单纯形, 怎样添加所需的新单纯形, 并确定简化规则。

当网格数据规模超大时, 部分研究者采用了分片载入处理的策略使得数据在现有计算框架之下得以处理。

参考文献

1. Garland M and Heckbert P. Surface Simplification Using Quadric Error metrics. In: SIGGRAPH 97 Conference Proceedings, Annual Conference Series, 209 ~ 216. ACM SIGGRAPH, Addison Wesley, August 1997

2. Paolo Cignoni, Claudio Rocchini, Claudio Montani, and Roberto Scopigno. External Memory Management and Simplification of Huge Meshes. IEEE Transactions on Visualization and Computer Graphics, 2003

(鲍虎军 张宏鑫)

sanwei xingtai huifu

三维形态恢复 (three dimensional shape recovering) 依据视觉计算理论, 在从二维图像恢复三维物体结构过程中, 求取物体可见表面的深度信息、表面朝向等 $2\frac{1}{2}$ 维描述的信息处理阶段。在这个阶段中, 各种不同信息 (用 X 表示) 可用相对独立的方法加以分析, 因而可以用不同的处理模块来恢复三维形态, 此过程的英语统称为 “shape from X ”。以下是一些典型的处理模块。

体视恢复深度 所用方法参见立体视觉。

影调恢复形态 物体表面的明暗变化 (影调) 可以给出三维信息, 如一些素描图上所涂阴影的明暗变化可以给人以立体感觉, 并能看出表面相对于光源的朝向。如图 1 所示, 物体上局部表面受光源照射后, 把一部分光线反射出来, 达到眼睛后就是我们感受到的明暗度 (在图像上就是灰度), 图中 n 是表面的法线方向, n_s 是光源的方向, n_o 是观测方向, i 是入射角, e 是反射角, g 是光源方向与观察方向间的夹角。表面上的灰度决定于许多因素, 其中包括光源强度、照射方向、物体表面材料的性质 (反射光的性质)、表面的朝向等。可以写出灰度与上述各因素间

的函数关系,因此在一定条件下就可以根据灰度计算出物体上某局部表面的朝向,这就是由明暗度恢复形态的基本原理。

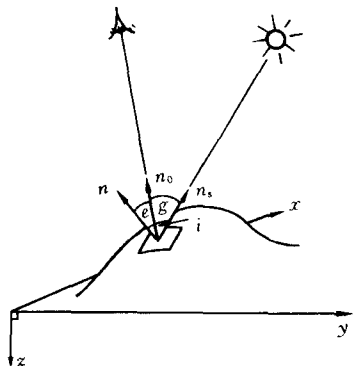


图1 局部表面对光照的反射

表面轮廓恢复表面朝向 根据表面上的轮廓线形状与透视投影原理可以恢复表面的朝向,主要过程如下:表面的朝向可以用两个度角 σ 与 τ 表示,如图2所示,物体表面上一点P,它在图像平面上的投影是P',n是P点局部表面的法线方向,n在图像平面上的投影是n',它与x轴的夹角记为 τ ,PP'与n的夹角记为 σ ,可用 (σ, τ) 表示P点局部表面的朝向。

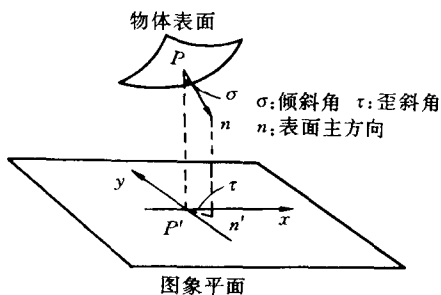


图2 表面朝向的表示

设物体表面是平面,其上的一轮廓线(图3)上有一点P,现研究P点对应的 (σ, τ) 与它在图像平面上投影点P'的相应角度间的关系。

过P作轮廓的切线,切线与 x' 轴的夹角为 β , x' 轴在图像面上投影为x轴。先假定 $\tau = 0$, β 角在图像面上投影为 β^* ,在图像面上,过P'点作轮廓投影的切线,那么此切线与x轴的夹角就是 β^* ,由图3可得

$$\tan \beta^* = \frac{\tan \beta}{\cos \sigma}, \quad \beta^* = \arctan \frac{\tan \beta}{\cos \sigma}$$

当 $\tau \neq 0$ 时,记 α^* 是图像中过P'点的轮廓投影的切线与x轴的夹角,则有

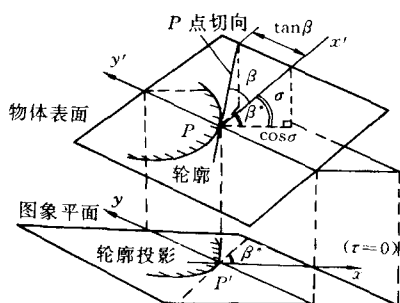


图3 轮廓上点与投影的关系

$$\alpha^* = \beta^* + \tau = \arctan \frac{\tan \beta}{\cos \sigma} + \tau$$

上式建立了图像面上物体表面上轮廓投影的切线方向与实际三维表面(局部表面很小时近似为平面)方向 (σ, τ) 之间的关系。处于图像面上的参数 α^* , β^* 等已知时,在一定条件下可以对 σ, τ 进行估计。

表面纹理(或称质地)恢复表面形态 纹理的概念已在图像分析中介绍过,图4显示出的纹理可提供有关表面朝向的信息(左、中图中箭头为表面法线方向,右图则向右和向后倾斜)。在提取出构成纹理模式的纹理基元后,如果对所有纹理基元的轮廓求出它所围的纹理基元的表面朝向,就可以得到物体表面朝向的分布。因此对于由明显基元构成的纹理型表面,上面讲的从轮廓估计表面朝向的方法也可用于由纹理估计表面的朝向。

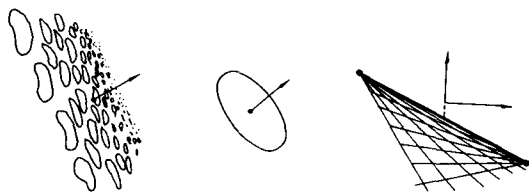


图4 纹理与表面朝向的关系

还有一种方法是从纹理梯度检测物体表面各点的相对距离。对具有相同纹理特征的物体表面作观测,可以发现,随着物体与相机光学中心距离的增大,纹理基元的尺寸变小,与视线平行的尺寸变小了。利用这一直觉,可以构造一种基于纹理梯度的测距方法,示意图如图5,其中S轴表示地面(纹理表面),Y轴为图像平面,O是焦点(相机光学中心),AB,BC,CD为观测窗口。各窗口中心存在一个纹理基元,它们的尺寸都是t; l_1, l_2, l_3 是AB,BC,CD窗口内纹理基元在成像面上的投影, $Y_A Y_B, Y_B Y_C, Y_C Y_D$ 是地面观测窗口在图像平面上的投影。它们的中心分

别是 Y_1, Y_2, Y_3 , 从图上可见 AB 最近, BC 居中, CD 最远, 因此, $l_1 > l_2 > l_3$ 。经过对几何关系的分析, 可以从图像平面上的一些参数求出 A, B, C, D 各点距离的相对值。

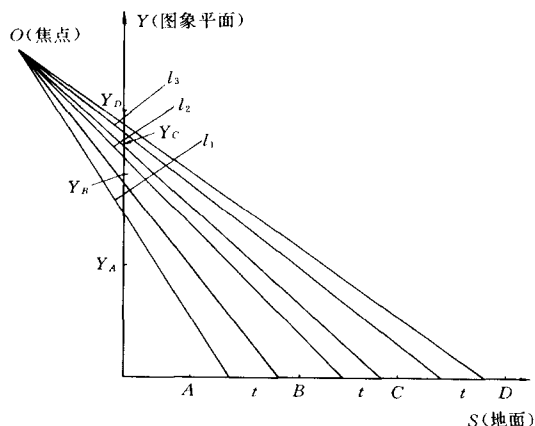


图5 从纹理梯度测距的几何模型

运动恢复三维结构 这是通过获取图像序列, 用光流场或特征对应关系恢复三维物体结构的方法, 参见运动分析。

参考文献

1. Marr D 著. 视觉计算理论. 姚国正等译. 北京: 科学出版社, 1988
2. 徐建华. 图像处理与分析. 北京: 科学出版社, 1992
3. 李介谷. 计算机视觉. 上海: 上海交通大学出版社, 1993 (阎平凡)

saomiaoyi

扫描仪 (scanner) 一种将图像信息输入计算机的设备。它将大面积的图像分割成条或块, 逐条或逐块依次扫描, 利用光电转换元件转换成数字信号并输入计算机。

扫描仪在计算机中的应用始于 1984 年。扫描仪由扫描头、控制电路和机械部件等组成。扫描头由光源、光敏元件和光学镜头等组成。光源通常采用长条状白色发光二极管或冷阴极管, 也有彩色扫描仪采用黄绿色发光二极管的。工作时照射到原稿 (即扫描对象) 上的光反射 (或透射) 到电荷耦合器件上, 电荷耦合器件本身是由许多单元组成的, 因此在接收光信号时将连续的图像分解成分离的点 (像素), 同时将不同强弱的亮度信号变成幅度不同的

电信号, 再经过模数转换成为数字信号。扫描完一行后, 控制电路和机械部件使扫描头或原稿移动一小段距离, 继续扫描下一行。扫描得到的数字信号以点阵的形式保存, 再使用文件编辑软件将它编辑成标准格式的文本, 存储在磁盘上, 以便进一步处理。一幅 300 dpi (点数每英寸) 的 A4 幅面的彩色图像, 最后形成的文本大小大约是 30 MB。

扫描仪种类很多, 可以按不同的标准来分类。按图像类型分有黑白、灰度和彩色扫描仪。按扫描对象幅面大小分可分为小幅面的手持式扫描仪、中等幅面的台式扫描仪和大幅面的工程图扫描仪。按扫描对象的材料分有扫描纸质材料的反射式扫描仪和扫描透明胶片材料的透射式扫描仪。按用途分除了通用的扫描仪外, 还有专用的扫描仪, 如卡片扫描仪、条码扫描仪等。

扫描仪是图像输入系统和文字识别系统中的重要设备。使用扫描仪可以将整幅信息直接输入计算机, 不但可输入文字, 还可输入图像、照片等, 大大提高了工作效率。在办公自动化领域, 扫描仪用于资料制作, 资料管理, 机械或其他工程图纸档案的管理等。此外, 扫描仪还用于模式识别, 如公安系统的指纹识别等。扫描仪也逐步进入家庭, 成为计算机重要的外部设备。

参考文献

- 郭繁夏. 扫描仪的原理与开发应用. 北京: 清华大学出版社, 1996 (林兼)

shaifa

筛法 (sieve method) 数论中有广泛应用的一个测试素数的初等方法。大约在公元前 250 年, 古希腊数学家 Eratosthenés 根据整数的如下性质, 提出求素数的方法。即, 如果 $n \leq N$, 而 n 是复合数, 则 n 必为一个不大于 \sqrt{N} 的素数所整除, 对于一个整数 N , 只要知道了不超过 \sqrt{N} 的所有素数, 就能求出不超过 N 的全部素数。具体做法是: 列出不超过 \sqrt{N} 的全体素数, 设为 $2 = p_1 < p_2 < \dots < p_r \leq \sqrt{N}$, 然后再列出 $2, 3, \dots, N$, 在其中留下 $p_1 = 2$ 而把 p_1 的倍数 $2p_1, 3p_1, \dots$ 全部划去, 再留下 p_2 而把 p_2 的倍数 $2p_2, 3p_2, \dots$ 全部划去, 如此继续下去, 直到最后留下 p_r 而划去 p_r 的全部倍数, 剩下的整数就是不超过 N 的全部素数。例如要求出不超过 50 的全部素数, 因为不超过 $\sqrt{50} < 8$ 的素数是 2, 3, 5, 7, 所以在 2, 3, \dots , 50 中留下 2, 3, 5, 7, 依次划去 2, 3, 5, 7 的倍数, 最后留下

的整数 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47 就是不超过 50 的全体素数。这就是著名的埃拉托斯特尼筛法。素数表都是根据这一方法略加变化而造出来的。

埃拉托斯特尼筛法的改进与发展,是近代解析数论的重要研究课题之一。许多古典数论问题,例如孪生素数猜想(即差为 2 的素数对有无穷多对),哥德巴赫猜想(即大于 2 的偶数可表为两个素数之和)都可用筛法研究,并且获得了部分成果。由 Ю. В. Лунийк 等人发展起来的大筛法已经有所进展。现在已经知道,存在一个整数 N ,使得若 p 取遍所有孪生素数,则所有它们的倒数 $1/p$ 之和小于 N ;而当 x 充分大时,可以使不超过 x 的所有这种素数 p 的相应和数任意大。1966 年,我国数学家陈景润证明了:所有偶数(除 2 以外)是一个素数和另一个整数之和,而后者或者是一个素数,或者仅仅是两个素数的乘积。同时,陈景润用类似方法也证明了有无穷多个素数 p ,使 $p+2$ 或者是素数,或者仅仅是两个素数的乘积。这是世界公认的筛法理论最卓越的应用成果,对近代筛法的进展有重要的影响。

参考文献

华罗庚. 数论导引. 北京: 科学出版社, 1957

(蒋增荣)

shangxiawen wuguan wenfa

上下文无关文法 (context free grammar)

形式语言理论中一种重要的变换文法,在乔姆斯基分层中称为 2 型文法,生成的语言称为上下文无关语言或 2 型语言,在程序设计语言的语法描述中有重要应用。

范式 上下文无关文法(CFG)可以化为两种简单的范式之一,即任一上下文无关语言(CFL)可用如下两种标准 CFG 的任意一种生成: 其一是乔姆斯基范式,它的产生式均取 $A \rightarrow BC$ 或 $A \rightarrow a$ 的形式; 其二是格雷巴赫范式,它的产生式均取 $A \rightarrow aBC$ 或 $A \rightarrow \alpha$ 的形式。其中 $A, B, C \in V$, 是非终结符; $a \in \Sigma$, 是终结符; $\alpha \in \Sigma^*$, 是终结符串。

歧义性和不确定性 从文法生成语言,可有多种推导方式。例如文法 $\{S \rightarrow AB, A \rightarrow a, B \rightarrow b\}$ 可有两种推导: $S \Rightarrow AB \Rightarrow aB \Rightarrow ab$ 及 $S \Rightarrow AB \Rightarrow Ab \Rightarrow ab$ 。若每次都取最左边的非终结符进行推导,如上例中的前一种方式那样,则称为左推导。如果有两种不同的左推导推出同一结果,则称此文法是歧义的,反之是无歧义文法。对有些歧义文法,可找到一个等价的

无歧义文法,生成同一个语言。不具有无歧义文法的语言称为本质歧义性语言。例如, $\{S \rightarrow A, S \rightarrow a, A \rightarrow a\}$ 是歧义的文法。 $L = \{a^m b^n c^n \mid m, n \geq 1\} \cup \{a^m b^n c^n \mid m, n \geq 1\}$ 是本质歧义性语言。接受 CFL 的自动机称为下推自动机。确定型和不确定型下推自动机接受的语言分别称为确定型 CFL 和不确定型 CFL,前者是后者的真子集。例如, $L = \{a^n b^n \mid n \geq 1\} \cup \{a^n b^{2n} \mid n \geq 1\}$ 是一个不确定型 CFL 而不是确定型 CFL。

同态映射下的性质 对任意正整数 n , 令 $\Sigma_n = \{a_1, \dots, a_n\}$, $\Sigma'_n = \{a'_1, \dots, a'_n\}$, 定义乔姆斯基变换文法 $G = (\Sigma, V, S, P)$ 为 $(\Sigma_n \cup \Sigma'_n, \{S\}, S, \{S \rightarrow, S a_i, S a'_i \mid 1 \leq i \leq n\})$ 。这个文法生成的语言称为代克集。如果把 a_i 看作开括号。把 a'_i 看作相应的闭括号,则 n 维代克集 D_n 就是由 n 种不同的括号对组成的配对序列之集合。例如, $a_1 a_2 a_2' a_2' a_1'$ 和 $a_1 a_1' a_2 a_2' a_1 a_1'$ 都属于 D_2 。

代克集是把正则语言族扩大成上下文无关语言族的工具。对任一上下文无关语言 L , 必存在两个同态映射 h_1 和 h_2 , 以及一个正则语言 R , 使 $L = h_2[h_1^{-1}(D_2) \cap R]$, 其中 D_2 是二维代克集, 反之亦然。

更进一步,上下文无关语言族是包含 D_2 , 且在同态、逆同态和与正则语言相交三种代数运算下封闭的最小语言族。关于其他的代数封闭性质见形式语言理论。

文法形式和文法的相似性 在符号置换的意义下(终结符和非终结符分别置换),许多文法之间有着相似性。把一组彼此相似的文法抽象成一个更高级的形式体系,称之为文法形式。迄今,文法形式的研究主要集中在上下文无关文法上。

文法形式的具体定义是: 给定无限的终结符表 Σ_∞ 和无限的非终结符表 V_∞ , 任取 Σ_∞ 和 V_∞ 的非空子集 Σ 和 V , 按构造普通文法的方法定义一个四元组 $G = (\Sigma, V, S, P)$ 。在 G 确定以后,任取映射函数 φ , 把 Σ 中每一元素 a 映射为 Σ_∞ 中一有限子集 $\varphi(a)$, 把 V 中每一元素 A 映射为 V_∞ 中一有限子集 $\varphi(A)$, 且当 $A \neq B$ 时有 $\varphi(A) \cap \varphi(B) = \emptyset$ 。 φ 就是所需的置换,通过它得到一个具体的文法 $\varphi(G) = [\varphi(\Sigma), \varphi(V), \varphi(S), \varphi(P)]$, 其中 $\varphi(P)$ 是把 P 中所有产生式中的符号作 φ 置换后得到的一组新产生式; $\varphi(\Sigma)$, $\varphi(V)$ 和 $\varphi(S)$ 分别是 $\varphi(P)$ 中出现的终结符集、非终结符集和出发符号。

这样的 G 称为文法形式, φ 称为 G 的一个解释,

$\varphi(G)$ 是 G 的一个解释文法,被认为是相似于 G 。令 φ 遍历各种可能的解释,得到的 $\varphi(G)$ 集合称为 G 的文法族,由此生成的语言集合 $\mathcal{L}(\varphi(G))$ 称为 G 的文法性语言族。例如,文法形式 $\{S \rightarrow aS, S \rightarrow a\}$ 的文法性语言族是正则语言集; $\{S \rightarrow SS, S \rightarrow a\}$ 的文法性语言族是上下文无关语言集。如果要求在 φ 置换时对 $a \neq b$ 有 $\varphi(a) \cap \varphi(b) = \emptyset$, 其中 a, b 为终结符,则在上述解释、解释文法、文法族、文法性语言族等名词前皆应加上“严格”二字。

若文法形式 G 作为普通文法时生成的语言 $L(G)$ 是无限集,则称 G 为非平凡的。此时文法性语言族 $\mathcal{L}(G)$ 是一个满主半 AFL (参见形式语言理论),反之不然。如满主半 AFL $\mathcal{L}(\{a^n b^n \mid n \geq 1\})$ 不是一个文法性语言族。

以 $\mathcal{L}_1 \cdot \mathcal{L}_2$ 表示文法性语言族 \mathcal{L}_1 和 \mathcal{L}_2 的乘积, $\mathcal{L}_1 \cup \mathcal{L}_2$ 表示两者之并,它们仍是文法性语言族。如果当 $\mathcal{L} \subseteq \mathcal{L}_1 \cdot \mathcal{L}_2$ 时,必有 $\mathcal{L} \subseteq \mathcal{L}_1$ 或 $\mathcal{L} \subseteq \mathcal{L}_2$ 成立,则称 \mathcal{L} 是素的。正则语言集和线性语言集都是素文法性语言族。任一文法性语言族 \mathcal{L} 必可惟一地分解为它的素因子乘积和: $\mathcal{L} = (\mathcal{L}_{i_1} \cdots \mathcal{L}_{i_{n_1}}) \cup \cdots \cup (\mathcal{L}_{m_1} \cdots \mathcal{L}_{m_{n_m}})$, 其中每个 \mathcal{L}_i 都是素因子。这里所说的惟一性是相对于求和运算 \cup 可交换而言的。

子文法类 可以根据不同的观点取上下文无关文法的子文法。一种观点是根据文法的外形和它们生成的语言族在代数运算下的封闭性。例如,若文法 G 的产生式只具有下列三种形式: $A \rightarrow \alpha\beta\beta, C \rightarrow \gamma$ 和 $S \rightarrow \delta$, 其中 $A, B, C \in V, \alpha\beta\gamma \in \Sigma^*, \delta \in (\Sigma \cup V)^*$, 且 δ 中至多含 k 个非终结符, S 是出发符号,且不在任何产生式的右部出现,则称 G 为 k 线性文法。1 线性文法又称线性文法。所有 k 线性文法统称元线性文法。元线性语言族在求并和乘积运算下是封闭的,但在求交、求补、乘幂闭包和置换等运算下皆不封闭。从包含关系说,正则语言族真包含于线性语言族。对任一 $k \geq 1, k$ 线性语言族真包含于 $k+1$ 线性语言族,元线性语言族真包含于上下文无关语言族。例如, $L_1 = \{a^i b^i \mid i \geq 1\}$ 是线性语言而不是正则语言。 $L_2 = \{a^{n_1} b a^{n_2} c a^{n_3} b a^{n_4} c \cdots a^{n_{k+1}} b a^{n_{k+1}+1} c \mid n_1 \geq 1, \cdots, n_{k+1} \geq 1\}$ 是 $k+1$ 线性语言而不是 k 线性语言。

由于上下文无关文法被广泛地应用于描述程序设计语言的语法,因此更重要的是从机械执行语法分解的角度取上下文无关文法的子文法,最重要的一类就是无歧义的上下文无关文法,因为无歧义性对于计算机语言的语法分解至为重要。在无歧义的上下文

无关文法中最重要的子类是 $LR(k)$ 文法,它只要求向前看 k 个符号即能作正确的自左至右语法分解。 $LR(k)$ 文法能描述所有的确定型上下文无关语言,但是对于任意的 $k > 1$,由 $LR(k)$ 文法生成的语言必可由一等价的 $LR(1)$ 文法生成。 $LR(0)$ 文法生成的语言类是 $LR(1)$ 文法生成的语言类的真子类。

语言识别的复杂性 Cocke-Younger-Kasami 三人分别独立地发现了一个算法,对任意给定的 CFG 和字符串 α ,它能识别 α 是否是该 CFG 生成的 CFL 中的句子。这个算法要用到乔姆斯基标准型。厄利发现了另一个具有同样识别功能的算法,但不需要把文法化成标准型。两个算法的时间复杂性都是 $O(n^3)$,空间复杂性都是 $O(n^2)$,其中 n 是 α 的长度。如果 CFG 是线性的,则可以修改上述算法,使时间复杂性降到 $O(n^2)$,空间复杂性降到 $O(n)$ 。如果 CFG 是无歧义的,则时间复杂性也可降到 $O(n^2)$ 。利用改进的厄利算法可以证明, $LR(k)$ 文法的语法分解复杂性在时间和空间上都是 $O(n)$ 。范利扬进一步把任意 CFG 的 CFL 识别时间复杂性降到 $O(n^{2.81})$ 。格雷巴赫发现了一个“最难识别”的 CFL,其含义是:如果有一个算法能在时间 $O(f(n))$ 和空间 $O(g(n))$ 内识别此语言,其中 $f(n) \geq n, g(n) \geq n$,则任何 CFL 均能在此时空限制下被识别。但迄今未能给出精确的 $f(n)$ 和 $g(n)$ 。

近年来开展了 CFL 并行识别复杂性的研究。鲁琢、李特等的算法可在 $O(\log^2 n)$ 时间内用 $O(n^6)$ 个处理器并行识别任意 CFL。若 CFL 是线性的,处理器数可降到 $O(n^3)$ 。若 CFL 是本质无歧义的,处理器数可降到 $O(n^2)$;或者只降到 $O(n^3)$,但同时使时间复杂性降到 $O(\log n)$ 。(陆汝钊)

shangxiawen youguan wenfa

上下文有关文法 (context sensitive grammar) 形式语言理论中的一种重要文法。一个四元组 $G = (\Sigma, V, S, P)$, 其中 Σ 是终结符的有限字母表, V 是非终结符的有限字母表, $S (\in V)$ 是开始符号, P 是生成式的有限非空集, P 中的生成式都为 $\alpha\beta\beta \rightarrow \alpha\gamma\beta$ 的形式, 这里 $A \in V, \alpha, \beta \in (\Sigma \cup V)^*, \gamma \in (\Sigma \cup V)^*$ 。上下文有关文法又称为 1 型文法。其生成式的直观意义是:在左有 α ,右有 β 的上下文中, A 可以被 γ 所替换。上下文有关文法所生成的语言称为上下文有关语言或 1 型语言。常用 \mathcal{L}_1 表示 1 型语言类。

单调文法 若文法 $G = (\Sigma, V, S, P)$ 的所有生成

式都为 $\alpha \rightarrow \beta$ 的形式并且 $|\alpha| \leq |\beta|$, 其中 $\alpha \in (\Sigma \cup V)^* V (\Sigma \cup V)^*$, $\beta \in (\Sigma \cup V)^*$, 则称 G 为单调文法。单调文法可简化得使 P 中任意生成式的右侧长最大为 2, 即: 若 $\alpha \rightarrow \beta \in P$, 则 $|\beta| \leq 2$ 。已经证明: 单调文法所生成的语言类与 1 型语言类, 即上下文有关语言类相同。因此, 有的文献把单调文法的定义作为上下文有关文法的定义。

例如: $G = (\{a, b, c\}, \{S, A, B\}, S, P)$, 其中: $P = \{S \rightarrow aSAB / aAB, BA \rightarrow AB, aA \rightarrow ab, bA \rightarrow bb, bB \rightarrow bc, CB \rightarrow cc\}$, 显然, G 是单调文法, 因而也是上下文有关文法。它所生成的语言 $L(G) = \{a^n b^n c^n | n \geq 1\}$ 是上下文有关语言。

上下文有关文法的标准型为: $A \rightarrow \xi, A \rightarrow BC, AB \rightarrow CD$, 其中 $\xi \in (\Sigma \cup V)$, $A, B, C, D \in V$ 。上下文有关语言类与线性有界自动机接受的语言类相同。1 型语言对运算的封闭性以及关于判定问题的一些结果参见短语文法中的表 1 和表 2。特别要指出的是: 1 型语言对补运算是否封闭是迄今未解决的一个问题。

参考文献

Hopcroft J E and Ullman J D. Introduction to Automata Theory, Languages and Computation. Addison-Wesley Publishing Company, 1979 (马世骅)

shebei ceshi

设备测试 (device testing) 为保证可靠工作,

而对计算机整机系统及其各部件与子系统进行的测试。设备测试主要对计算机系统的各种设备如打印机、软磁盘驱动器、硬磁盘驱动器以及显示器等的功能、机械特性、电气特性等根据事先制定的技术规范、测试标准、参数范围等进行严格的测试。测试采用专用或通用测试仪器与计量工具来实现, 以保证受测设备的可靠性、互换性、稳定性和其他质量指标。

设备测试一般可分为制造过程中对关键零部件的测试及装配后对整个设备的测试。关键零部件的测试又可分为对机械部件的测试与电气部件的测试。整个设备的测试又可分为单项测试与综合测试, 综合测试一般有自检和联机测试。

上述这些测试, 其测试的内容、方法、所用测试仪器设备、测试的环境条件等, 随设备的不同而有所不同。

(周学仁)

shebeijian zixitong

设备间子系统 (equipment subsystem) 结构化布线系统中安装在设备间内的布线子系统。设备间是指集中安装大型通信设备的场所, 如专用自动小交换机、大型计算机、计算机网络通信中枢等设备。并非每一个结构化布线系统都有设备间子系统, 但在大型建筑物中一般是有的, 而且有时还不止一个。设备间子系统如图 1 所示。

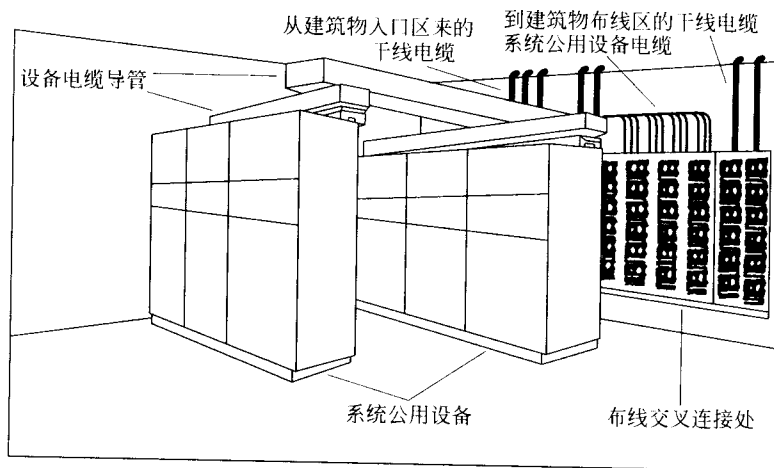


图 1 典型的设备间子系统

设备间子系统中的电话、终端设备、计算机主机设备及其保安配线设备宜设在一个房内。必要时, 也可以分别设置, 但程控交换机及计算机主机房离

设备间不宜太远。

设备间位置及大小应根据设备的数量、规模以及最佳网络中心布局要求, 综合考虑确定。

在设备间子系统的设计和安装过程中还需要综合考虑配电系统(不间断电源 UPS)和安全因素(设备接地等)。

参考文献

胡道元主编. 网络设计师教程. 北京: 清华大学出版社, 2001 (王晓东)

sheji gongju

设计工具 (designing tool) 软件开发过程中用于设计活动的工具, 它辅助设计人员从软件功能规约出发, 得到相应的设计规约。

设计工具可分为**概要设计工具**和**详细设计工具**, 概要设计工具用以设计目标软件系统的体系结构、控制结构和数据结构, 通常采用模块结构图来表示目标系统的结构, 其基本构成是模块与模块调用。概要设计工具除应对模块图结构的一致性及其完备性进行检测外, 还应提供对模块结构图质量的评价, 以便软件设计人员在多种设计方案中找到满足期望目标的设计方案, 得到可供详细设计的设计规约, 其典型代表有结构化设计工具等。详细设计工具用以设计模块内部的实现细节, 通常采用的表示形式有层次输入-处理-输出 (HIPO) 图、程序设计语言 (PDL)、问题分析 (PAD) 图等, 其典型代表有 PDL 设计工具等。

在设计阶段, **数据词典管理工具**保存了设计中各对象的相互联系和使用情况, 并在设计过程中维护命名空间, 提供相应的定义、更新及检索功能。

除了基于图形表示设计规约的设计工具外, 还有一类采用某种形式化功能规约描述语言的设计工具。它用以辅助设计人员建立目标系统的形式化设计模型, 并可对设计模型进行分析和验证。现已有一些实验性系统出现, 典型工具有可达性分析工具、模型检查工具、约束分析工具等。

可达性分析工具通过枚举一个系统的所有可能输入状态来构造出系统的可达性图。对此图的分析可得出一些有用结果, 如潜在的灾难性状态是否可达, 系统是否可能陷入死锁, 在同样条件下系统是否会显示不同的行为特征等。模型检查工具接受一个表示系统可达性图的逻辑模型和一组表示设计目标的逻辑公式, 来判断规约是否达到要求。约束分析工具用来验证系统的行为规约是否可满足, 即把控制流等规约转换成一组线性不等式, 如果它们有解, 则规约是可满足的。

面向对象设计 (OOD) 工具是 90 年代设计工具

的重要发展方向之一。此外, 设计工具发展方向还包括, 更多地表达软件设计活动本身的信息以提高软件设计重用程度和把多个不同的设计工具在共同的数据表示基础上集成为设计工具集以覆盖整个软件设计活动。

参考文献

1. Schindler M. Computer-Aided Software Design: Building Quality Software With CASE. New York: John Wiley & Sons Inc., 1990

2. Stevens W. Software Design: Concepts and Methods. Englewood Cliffs: Prentice - Hall Inc., 1991

3. Rumbaugh J, et al. Object-Oriented Modeling and Design. Englewood Cliffs: Prentice - Hall Inc. 1991

(钱乐秋 赵文耘)

sheji guiyue

设计规约 (design specification) 对软件的组织或其组成部分的内部结构的描述, 该系统满足给定的**功能规约**与需求定义所指定的全部功能及性能要求。通常有概要设计规约和详细设计规约, 分别为相应设计过程的输出文档。

概要设计规约 概要设计规约指明软件的组织结构, 其内容包括:

(1) 系统环境 硬件、软件接口与人机界面; 外部定义的数据库; 与设计有关的主要限定条件。

(2) 设计描述 数据流和主要数据结构; 软件的模块结构; 模块之间的接口。

(3) 对每个模块的描述 处理过程; 界面定义; 数据结构; 必要的注释。

(4) 文件结构和全局数据 文件的逻辑结构、记录描述以及访问方式; 全局数据定义; 交叉引用信息。

此外, 还应包括有关软件测试等方面的要求与说明。

概要设计规约描述软件的功能模块及其相互关系, 说明处理过程的外部行为, 但不指定过程算法。它对数据的描述通常限于主要数据对象的逻辑结构。概要设计规约是面向软件开发者的文档, 主要作为项目管理人员、系统分析人员与设计人员之间交流的媒介。

详细设计规约 详细设计规约描述软件各组成部分的内部结构, 它是概要设计规约的细化, 在概要设计规约的基础上加入以下内容:

(1) 各处理过程的算法。

(2) 算法所涉及的全部数据结构的描述,对主要数据结构往往包括与实现有关的描述。

详细设计规约主要作为软件设计人员与程序人员之间交流的媒介。

在概要设计规约与详细设计规约的内容之间应有明确的界限。详细设计中对系统不同部分的细化程度也可能有明显的差别。软件开发环境的发展对详细设计规约的内容有很大影响。

设计规约文档的形式 完整的设计规约包含多项文档,从各个方面对软件进行描述。目前使用最广泛的形式仍是具有特定形式的图形加上形式或非形式的正文。大多数情况下,概要设计规约与详细设计规约有相似的形式,差别在正文的内容。可选择的这类文档形式主要有:

(1) 定义逻辑模块的方框图 这里的逻辑模块可以描述抽象数据类型或数据对象,也可以表示功能。

(2) 实体-联系图(E-R图) E-R图着重反映要处理的信息的结构特征,因此常用于描述大型信息系统中的数据库部分。

(3) 数据流 能反映软件中修改、读写、删除、移动数据的动态过程。适于描述并行或分布式系统的 Petri 网也属于数据流类。

(4) 控制流 将软件看作若干基本操作的组合,组合方式由控制转移来确定。除广泛使用的流程图外,控制流也有各种非图形描述手段,如类似程序的 PDL 语言描述、含抽象级较高的成分的语言,如 SETL 语言等。

(5) 状态转换图 描述软件各组成部分之间的相互交互或与外部交互的部分。它和程序之间有较好的对应关系。

近年来形式设计规约受到广泛注意。一些形式化的语言把书写设计规约作为目标,它们主要采用代数或一阶逻辑等数学手段,提供抽象级别较高的模块定义与抽象数据类型机制(参见 SETL 语言, OBJ 语言, Larch 语言)。

一个软件的设计规约要涉及不同形式的设计文档,由于着重点的不同形成不同的设计途径,面向对象的设计是基于实体的,与传统的基于功能分解途径有很大区别(参见面向对象程序设计)。

参考文献

1. Freeman P, Wasserman A I (Eds.). Software Design Technique. 4th ed. Silver Spring: IEEE Computer Society Press, 1983

2. Fairley R E. Software Engineering Concepts. New York: McGraw-Hill, 1985 (陈道蓄)

shejixing yuyan

设计性语言 (design language) 用于书写软件设计规约的语言。设计性语言用于软件的概要设计及详细设计,其描述对象为软件系统的组织或其组成部分的内部结构,不同于重在定义软件系统及其组成部分的界面特性的功能语言。它也并不描述一个可以高效执行的软件的全部细节,因此有别于用于编程的程序设计语言。

设计性语言是软件设计人员、项目管理人员和程序实现人员之间交流的工具。一个好的设计性语言应该包含已知最有效的控制及数据结构概念,以便简明、自然地描述软件实现策略,并能在不同的程度上省略细节,可以在不列出低层细节的条件下明确地描述算法、数据结构和子任务的基本特性。

发展过程

早期使用图形化或半图形化设计性语言,并插入用自然语言书写的正文描述。后来出现了非形式化的设计性语言,广泛地用于书写设计规约。目前结构设计中使用的仍然以非形式化的语言为主,但形式化的设计性语言受到越来越多的注意,特别是在详细设计中应用较多。

主要类型

以下简要介绍目前使用的几种主要的设计性语言,每类语言的基本成分可参见各相关条目。

图形化的设计性语言 图形化的设计描述工具有多种,广泛使用的有流程图、盒图(N-S图)和问题分析图(PAD)等。流程图的基本成分很简单,描述方式自然,既可用于概要设计,也可用于详细设计。流程图可以支持结构程序设计,但它本身并不提供避免良好结构被破坏的机制。盒图则有此优点。盒图功能作用域明确;控制转移受严格控制;局部或全程数据的作用域容易确定;容易描述子程序递归调用。开发针对盒图的计算机辅助工具较困难,其使用受到很大限制。日本日立公司于1979年提出的问题分析图则特别适合于计算机支持下的详细设计。对应于不同的常用程序设计语言,可以有不同风格的PAD图。利用各自的对应表,能用比较机械的方法将图转换为程序。

表格形式的设计性语言 在许多应用系统中,每个模块的功能表现为对一个复合条件求值,根据

结果选择适当的动作。对于这样的系统,可以用决策表作为设计描述工具。决策表早在软件工程出现之前就有人使用,随着软件工程技术的发展,现在已有对决策表进行完备性、一致性验证和化简的工具;并且出现了直接以决策表为输入的表驱动算法。因此,决策表已经成为一种有效的过程设计语言。

设计性程序语言 这类语言书写的文档具有和实现性语言类似的结构,但控制结构内部的细节描述采用自然语言,因此这类语言可以认为是结构化英语,属于非形式化语言。

为了便于进行从设计到实现的转换,经常用各种高级语言作为相应设计性语言的基础,有 Ada-PDL, Pascal - PDL 等各种适合于不同开发环境的设计性语言。设计性程序语言应该具有如下特征:

(1) 文法有固定的关键字,以提供描述所有结构化构造、数据说明以及模块特性的机制;

(2) 一般用自然语言;

(3) 可以定义简单和复合数据结构,包括抽象程度较高的数据结构,如链表、树等;

(4) 可以描述子程序定义和能支持各种界面描述方式的调用。

为满足以上要求,基本设计程序语言的语法应包括:子程序定义、界面描述、数据说明和类型定义、块结构、条件构件、循环构件、I/O 构件。设计性程序语言(PDL)这个名词除指一类设计性语言外,还经常指由 S. Caine 等人提出的一种特定的设计性语言,这种 PDL 是使用最多的设计性语言之一。其格式和示例参见 PDL 语言。

按照传统的观点,设计性程序语言应当是可以扩充的;一方面通过扩充包容控制与数据结构方面的新概念,如多任务、并行处理、进程间通信、声象界面等;另一方面,通过扩充支持不同应用领域特定的结构。然而,近来的倾向是让语言有固定的结构。一则是因为 CASE 技术的发展提高了对设计过程自动支持的程度,同时也要求语言有严格的文法。因此,语言中必须包括有足够表达力的成分,能适应不断发展的应用要求,诸如用户定义抽象数据类型、类属模块、多继承机制、并行循环、非确定等待、进程的条件激活机制等。而领域特定结构则可以通过领域专用库来提供。

形式化的设计性语言 图形与表格语言中的正文部分几乎都使用自然语言,而程序设计语言本质上仍是自然语言。因此,它们都有歧义性。要解决这个问题,应使用形式化的设计性语言。目前,在

详细设计中使用代数语言等描述数据结构定义,或者在实现性语言中加入设计级的成分,而功能性语言中有的也含有设计级成分(参见**功能性语言**、**Z 语言**、**FGSPEC 语言**)。

参考文献

1. Martin J, McClur C S. Diagramming Techniques for Analyst and Programmers. Englewood Cliffs: Prentice - Hall, 1985

2. Linger R C, Mills H D, Witt B I. Structured Programming — Theory and Practice. Reading: Addison-Wesley, 1979 (陈道蓄)

shexiangji biaoding

摄像机标定 (camera calibration) 确定摄像机固有的光电和几何参数(统称为内参数)的过程和方法。

摄像机标定是从二维图像恢复三维空间几何结构必不可少的步骤,是三维计算机视觉的重要研究内容。由于摄像机制造厂家提供的内参数一般来说不能满足应用精度的需求,所以需要对摄像机进行标定。摄像机标定可以分为传统标定和自标定两大类。传统标定是指在标定过程中使用定标块的标定方法。定标块是指加工精度很高且结构尺寸已知的一个基准物体。自标定是指不需要定标块、仅利用多幅图像间几何基元(如点、线等)之间的对应关系进行标定的方法。自标定本质上利用的是射影空间无穷远平面上的绝对二次曲线或绝对二次曲面在图像平面上的像与摄像机的运动无关、仅与摄像机的内参数有关的事实。摄像机标定一般是指对针孔模型下摄像机内参数的确定过程。在精度要求很高的应用场合,需要对摄像机的非线性畸变参数同时进行标定,摄像机最主要的畸变参数是径向畸变参数。

(胡占义)

shenhuajia cunchuqi

砷化镓存储器 (GaAs memory) 采用砷化镓半导体存储器件和外围电路组成的高速存储器。按其存储方式分为高速随机存储器与快速串行电荷耦合存储器两种类型。

砷化镓高速随机存储器 砷化镓电子迁移率高(约为一般硅材料的 7 倍),且材料的本征电阻率大(约 $10^9 \Omega \cdot \text{cm}$),有利于提高存储单元的开关速度。砷化镓半导体材料禁带宽,工作温度高,抗辐射能力

强。GaAs 器件在速度和集成度方面均可达到射极耦合逻辑 (ECL) 器件的水平,而功耗比 ECL 器件低。

GaAs 储存信息的基本单元是由砷化镓-金属-半导体场效应管 (GaAs MESFET) 构成。GaAs MESFET 的工作原理与结构同硅场效应管 (Si FET) 相似。与金属-氧化物-半导体场效应管 (MOSFET) 的差别在于它的肖特基二极管是在金属栅和 GaAs 界面处形成,可提高沟道电流,进而提高存储的速度。

MESFET 有耗尽型和增强型两类。耗尽型在零栅压条件下沟道仍导通,漂移大,大规模集成存储单元难度极大。增强型在零栅压条件下,依靠其栅极 PN 结或肖特基势垒的自建势,使沟道夹断,没有导通电流,当加以正压 (N 沟道器件) 补偿其自建势垒,达到某一阈值时,沟道才开始导通。因此器件抗干扰、抗辐射能力强。高速 GaAs 存储单元大多数使用增强型 MESFET 直接耦合逻辑构成。

一般 GaAs 静态随机存取存储器 (SRAM) 逻辑电路采用直接耦合场效应管逻辑 (DCFL),虽然逻辑电平摆幅小,但功耗低。这种直接耦合逻辑阵列,可由用户根据存储单元安排进行设计布线。DCFL 采用细长金属栅结构,这种结构漏电流小,有利于提高集成度。富士通公司生产的 GG 11/12 砷化镓门阵列,门延迟为 50 ps,在每块芯片装有 10 kb 高速 RAM 门阵列,存取速度达 2 ns。

GaAs 半导体制成的高速静态随机存取存储器可用于高性能的精简指令集 (RISC) 计算机作为高速缓冲存储器 (cache),它可采用嵌入方式和中央处理器 (CPU) 紧密结合,避免走线交叉延时,从而提高了速度。

砷化镓快速串行电荷耦合存储器 砷化镓半导体器件工作频率高,电荷转移效率可达 99.99%,是理想的电荷耦合器件。目前已制成 3.2×10^5 位高集成度的电荷耦合器件存储器。它可用于图像信息的存储,也是未来光计算机的重要存储器件。

近年来由于半导体量子光电子学技术的进展,可生长一种人工改性的超微结构的半导体材料,即量子阱超晶格材料,如砷化镓异质结材料 AlGaAs/GaAs/AlGaAs 等,这种砷化镓材料有较好的光电子特性,它和电荷耦合存储技术相结合,可以构造较理想的图像信息存储器,这对于光字符识别、模式识别将起很大作用。

参考文献

1. 林雨. 半导体存储器及其测试. 北京: 科学

出版社, 1980

2. Chandna A, Brown R B. A 32 Kbit GaAs SRAM with Electrically Programmable Redundancy. Proc. 1993 Symp. Research Integrated System

(黄德全)

shenhuaqia jicheng dianlu

砷化镓集成电路 (GaAs integrated circuit, GaAs IC) 采用砷化镓一类半导体材料制造的集成电路。这种材料是 III-V 族化合物中最典型的一类半导体材料。

与最为常用的硅半导体集成电路相比较,砷化镓集成电路具有运算速度快、功耗低、工作电压低、输入阻抗高、反向隔离度大、集成度高、抗辐射能力强和制造的工艺流程简单等优点。砷化镓集成电路已成为超高速集成电路 (VHSIC) 的专用名字,是近年来最为关注的、发展最迅速的超高速集成电路,但是目前它的产量和价格还远远不及硅集成电路。用砷化镓制成的各种门电路、触发器、分频器到大规模运算电路、存储器、门阵列、数字转换电路等都已实用化。作为无线发射、接收领域的应用,例如卫星广播电视通信、蓝牙技术、无线网络、全球个人移动通信、空中及陆地交通管理等,都离不开砷化镓超高速集成电路。

由于分子束外延 (MBE)、金属氧化物化学气相淀积 (MOCVD) 等材料的生长技术与电子束 (EB) 等微细加工技术的出现,诞生了 GaAs, InP 等异质结高电子迁移率器件。其中有高电子迁移率晶体管 (HEMT)、伪晶高电子迁移率晶体管 (PHEMT)、异质结双极晶体管 (HBT)、异质结场效应晶体管 (HFET) 等开关器件及毫米波单片集成电路 (MMIC)。砷化镓超高速集成电路的门延迟时间小于 1 ns 或工作频率大于 1 GHz。可用砷化镓门电路和触发器等构成与硅集成电路相类似的电路品种。

砷化镓超高速集成电路的逻辑形式

常用的逻辑形式有直接耦合场效应晶体管逻辑 (DCFL), 缓冲场效应晶体管逻辑 (BFL), 源耦合场效应晶体管逻辑 (SCFL), 肖特基二极管场效应晶体管逻辑 (SDFL), 低夹断电压场效应晶体管逻辑 (LP-FL), 电荷耦合场效应晶体管逻辑 (CCFL) 以及互补电平场效应晶体管逻辑 (CLFL) 等。

直接耦合场效应晶体管逻辑 (DCFL) 采用增强型场效应晶体管作为开关元件, 耗尽型场效应晶体管或电阻作为负载, 具有单电源、低功耗、电路结构

简单、易实现和集成度高的优点。这种逻辑形式在砷化镓大规模集成电路设计中最为常用。其缺点是电平摆幅小,噪声容限低,要求电路中的场效应晶体管阈值偏差要小,这对砷化镓材料制备工艺和电路制造技术提出了较高的要求。

砷化镓超高速集成电路(VHSIC)最早采用的逻辑形式是缓冲场效应晶体管逻辑(BFL)。VHSIC使用耗尽型场效应晶体管,由双电源供电,带有输出缓冲级以实现电平移位和提高驱动能力。其优点是电路速度快,驱动能力强,输出逻辑的摆幅大,噪声容限大,并且对电路中的晶体管阈值偏差要求比较宽松,制造工艺简单等。这种电路由双电源供电,耗尽型器件功率损耗又大,因此在中小规模集成电路中使用较多。

源耦合场效应晶体管逻辑(SCFL)可采用增强型或耗尽型场效应晶体管。增强型场效应器件功耗低,集成度高,工艺难度比DCFL要小。由于这种逻辑电路的输出逻辑摆幅与阈值电压关系不大,且因具有互补输出而易构成触发器以及时钟频率较高等优点,已在各种规模的砷化镓超高速集成电路中得到广泛的应用。

肖特基二极管场效应晶体管逻辑(SDFL)电路采用耗尽型场效应晶体管,并由两路电源供电,扇出和驱动能力不强,工作速度也不如上述三种电路。

互补电平场效应晶体管逻辑(CLFL)是一种新的逻辑电路。它在DCFL的基础上采用互补信号的输入和输出。电路模拟指出,在相同速度下其功耗可比DCFL降低5~500倍,因而适用作低功耗电路。

砷化镓超高速集成电路的逻辑元件

砷化镓超高速门电路 砷化镓超高速集成电路中最简单、最基本的电路,由它可构成各种功能的超高速集成电路。砷化镓超高速门电路的传输延迟时间达100~500 ps,输出波形的上升和下降时间为100~200 ps。

砷化镓超高速D型触发器 它通常采用DCFL、BFL和SCFL的逻辑形式,是砷化镓超高速集成电路的基本电路。它具有50 Ω 负载的驱动能力,工作频率达8~10 GHz,输出波形的上升和下降时间为100~200 ps,建立和保持时间为100~150 ps。

砷化镓超高速分频器 通常又分为静态和动态两种分频器。静态分频器由门电路和触发器构成,动态分频器也称预定标器,多由BFL和传输门组成。静态分频器最高频率已达20~30 GHz。一种用

HEMT器件采用DCFL电路制成的1/8动态分频器其工作频率达26.5 GHz,功率为573 mW。

砷化镓存储器电路 砷化镓超高速集成电路在向大规模集成电路发展的过程中,首先研制的是静态随机存取存储器(SRAM),且主要采用功耗较小的DCFL电路。原因是其成本较高,存储容量也有待加大。

砷化镓门阵列和标准单元 砷化镓门阵列产品已达万门级水平。一种3万门GaAs MESFET采用SCFL逻辑形式的门阵列的门延迟时间为90 ps,功耗仅为硅ECL的1/4。GaAs高速门阵列产品近年发展很快。

砷化镓微处理器 用单片集成技术制作在GaAs衬底上的中央处理器与用硅材料的相比,砷化镓微处理器具有高速、低功耗的特点。

砷化镓运算器 用砷化镓材料制成的加法器和减法器与用硅材料的相比,其运算速度要快得多。例如,用HEMT器件制成的16×16复数乘法器用22 000个有源器件构成4 500个门电路,工作频率达520 MHz,功耗为4 W。

砷化镓模数和数模转换器 大多采用SCFL逻辑形式。已有14位、工作频率为2 GHz、功耗为2.5 W的数模转换器产品。

砷化镓分布式放大器 它在带宽、性能和芯片尺寸方面有优越性。已有5~100 GHz的5级7 dB的砷化镓分布式放大器,单位面积增益达到1 dB/mm²。

砷化镓功率MMIC 砷化镓功率MMIC的性能可达:12.0~16.0 GHz,2 W,18 dB;42.5 GHz,0.5 W,15.1 dB。

砷化镓毫米波PHEMT 砷化镓毫米波PHEMT的性能可达:16 GHz,7 W,4.5 dB;55 GHz,219 mW,4.1 dB。

由于MBE、MOCVD等能生长几个分子层厚度的薄膜材料的设备与电子束、反应等离子体蚀刻(RIE)微细加工技术的出现,诞生了以Ⅲ-V族化合物GaAs为主体的、只有几个分子层厚度的各种材料的薄层结构,例如AlGaAs/GaAs, InGaAs/GaAs, InAlAs/InGaAs/InP。再加上HEMT, HBT和弹道式收集晶体管(BCT)的研制成功,将使GaAs超高速集成电路向工作速度更快、功率更大、损耗更小的方向发展。

参考文献

1. Wing O. Gallium Arsenide Digital Circuits.

Boston: Kluwer Academy Publishers, 1990

2. 李效白. 砷化镓微波功率场效应晶体管及其集成电路. 北京: 科学出版社, 1998

(汪锁发 杨玉芬)

shenjing jisuan

神经计算 (neural computing) 在人工智能中, 研究人工神经网络的建模与信息处理的分支。神经计算通过对人脑的基本单元(神经元)的建模和联结, 来探索模拟人脑神经系统功能的模型, 研究非程序的、适应性的、大脑风格的人工神经网络信息处理的本质和能力, 并研制一种具有学习、联想、记忆和模式识别等智能信息处理功能的人工系统。人工神经网络是由大量处理单元互连组成的非线性、自适应的动力系统, 是一种并行分布式系统, 它采用了与传统人工智能和信息处理技术完全不同的机理, 克服了传统基于逻辑符号人工智能在处理直觉、非结构化信息方面的缺陷, 具有自适应、自组织和实时学习的特点。

1943 年心理学家 W. S. McCulloch 和数理逻辑学家 W. Pitts 建立的 MP 神经网络模型, 开创了神经计算的研究。1969 年 M. Minsky 和 S. Papert 在“Perceptron”一书中证明了感知机的局限性后, 神经计算一直处于低谷状态。直到 1982 年美国物理学家 J. J. Hopfield 提出了 Hopfield 神经网络模型, 神经计算的研究才又活跃起来。

神经计算主要研究神经网络模型, 包括网络连接的拓扑结构、神经元的特征、学习算法等。目前, 已有近 50 种神经网络模型, 其中有反传网络、感知机、自组织映射、Hopfield 网络、玻耳兹曼机、适应谐振理论等。根据连接的拓扑结构, 神经网络模型可以分为前向网络和反馈网络两种。人工神经网络所具有的适应性行为通过学习实现, 即根据环境的变化, 对权值进行调整, 从而使系统的行为得到改善。神经网络学习算法的基础是 Hebb 学习规则, 由加拿大心理学家 D. Hebb 在《行为的组织》一书中提出, 学习导致突触的联系强度和传递效能的提高, 即为 Hebb 学习规则。在此基础上, 人们提出了各种学习规则和算法, 以适应不同网络模型的需要。有效的学习算法, 使得神经网络能够通过连接权值的调整, 构造客观世界的内在表示, 形成具有特色的信息处理方法, 信息存储和处理体现在网络的连接中。人工神经网络已在模式识别、智能控制、组合优化、预测等领域获得了成功的应用。

参考文献

史忠植. 神经计算. 北京: 电子工业出版社, 1993 (史忠植)

shenjing jisuanji

神经计算机 (neural computer) 一种在生物神经元数学模型的基础上构造人工神经网络 (ANN) 以实现计算和信息处理的非传统计算机。又称神经处理器。神经计算机的组成原理和工作方式完全不同于传统计算机。后者是模拟人类基于符号化概念的抽象思维的信息处理功能, 而神经处理器则是在神经网络微观结构这一层次上模拟其对刺激-反应的自适应调整来进行信息处理。

神经计算实质上是 ANN (参见人工神经网络) 在经过学习或训练后所表现出来的对输入信息的适应性反应。但是, ANN 毕竟是生物神经网络的高度简化模型, 无论是数量上还是品质上都远远比不上生物神经网络, 尤其是无法与人脑神经系统相比拟。通常是针对某一特定问题设计一个 ANN 并加以训练后, 用于求解该特定问题。比较现实的研究开发途径是把神经处理器作为协处理器, 在嵌入式系统或控制系统中发挥其特定作用。

训练 ANN 时所要进行的数学运算, 主要是连接权值调节量和各单元多路输入信号量的矩阵乘、加计算, 而且相对独立的单元及其连接可以并行处理。神经计算机主要有以下 3 种实现技术:

(1) 计算机软件仿真 通过编程, 在通用计算机上实现某种学习算法来建立并训练一个“软”神经网络。

(2) 用专用并行处理器器件进行硬件仿真 用这样的器件组成专用协处理器将显著加速“软”神经网络的学习速度, 缩短其训练时间。

(3) 用线性集成电路器件 直接将电子神经元模拟线路组成的 ANN 制作成集成电路的神经处理器芯片。这要根据特定的用途, 利用 ANN 开发工具在通用计算机上训练好“软”神经网络, 取得有关数据, 再根据这些数据设计 VLSI 掩模, 制作专用芯片。由于线性放大器电路的 VLSI 芯片集成度难以提高, 所以主要是针对特定用途, 生产一些集成度适中的产品。

ANN 的研究开辟了一条不同于传统计算机的信息处理途径。尤其是在涉及自动控制、模式识别、人工智能等应用方面, 已取得了不少用传统计算机难以达到同样效果的成果, 包括文字识别、语音识

别、目标辨识、自然语言理解、专家系统(例如,金融市场预测和观测数据回归分析等)及过程控制等。

参考文献

1. Wasserman P D. Neural Computing—Theory and Practice. New York: Van Nostrand Reinhold, 1989
2. 史忠植. 神经计算. 北京: 电子工业出版社, 1993
3. 施鸿宝. 神经网络及其应用. 西安: 西安交通大学出版社, 1993 (童颖)

shenjing wangluo xinpian

神经网络芯片 (neural network chip) 模拟脑神经网络生物信息处理过程中的某些特征而制成的一种集成电路。它是构成神经计算机的基础元器件,目前的神经器件大多数利用微电子大规模集成技术和光技术来实现。

按照信息主要是由电子还是由光子来传输,神经网络芯片可分为电子神经芯片和光子(光电)神经芯片。电子神经芯片通常以硅 CMOS 集成电路工艺实现,故也称硅神经器件;光子神经芯片则以化合物半导体光子或光电集成电路工艺实现。

硅神经器件通用性较强,精度高,速度快,已初步达到实用。按功能分类,硅神经器件分为基于 Hopfield 模型和 BP 算法,能模拟人脑“记忆、学习”功能的器件,以及能模拟生物“感觉”器官的器件,如模拟生物视觉系统的硅视网膜和模拟生物听觉系统的硅耳蜗。硅神经器件的实现形式有模拟与数字两类。模拟电路技术比较成熟,电路简单,特性与神经网络更为接近,但功耗较大,而且用可变电阻表示互连强度,工艺实现难度较大。通过改变可变电阻数值来调整互连权重值可以实现学习功能。数字电路容易集成,不需要高阻值的电阻和大电容,缺点是电路复杂,所需门电路数目较多。目前已研制出包含 512 个神经元的硅神经器件。规模更大时,由于互连问题而十分困难,但仍有可能达到 1 000 个神经元(相当于 40 万个门电路)的水平。

光子神经芯片利用高密度的光子互连技术制成。由于光波传播无交叉失真,传播容量大,运算速度快,在实现神经网络的设计方面有明显的优势。目前光子神经芯片用于实现人脑联想记忆功能和视网膜。光子神经芯片的实现形式有发光二极管阵列和光探测器阵列芯片构成的基于 Hopfield 模型的交叉棒结构,以及由发光二极管、异质结晶体管或金属-半导体场效应晶体管(MES-

FET),光电二极管三者集成的光双稳面阵列结构。光子神经芯片与存放互连权重值的掩模版或全息片一起实现求积和运算以及取阈功能。利用时分多路复用技术,可以大大减少元件的数目。用空间光调制器来改变互连权重值,或改变全息片可以实现学习功能。研究表明光双稳阵列的集成度可达 10 000 个神经元/cm²。一些厂家重视交叉棒结构,他们提出的可变灵敏度光电探测器阵列能完成空间光调制器和光电探测器两者的作用,并实现了学习功能,省略了光学元件,结构大为简化。这是一种兼含微电子可集成性和光电子高度互连性优点的光电神经芯片,预计集成度可达 2 000 个神经元/cm²以上。一种光电神经网络系统已能实时识别别人的面孔,正在走向实用阶段。

神经网络计算机和专用神经网络芯片的研究将促进新一代具有人工智能的生产工具和武器的发展,具有十分重要的应用前景。

参考文献

- 庄镇泉,王熙法,王东生. 神经网络与神经计算机. 北京: 科学出版社, 1992 (夏永伟)

shenxiao koujing tianxian diqiuzhan

甚小口径天线地球站 (very small aperture terminal, VSAT) 一类具有甚小口径天线,价格低廉,可以方便地安装在用户场地的智能化小型或微型地球站。一般由大量甚小口径天线地球站(VSAT)小站与一个主站协同工作,共同构成一个广域稀路由(站多,各站业务量小)的 VSAT 卫星通信网。

与地面通信网相比,VSAT 卫星通信网具有以下特点:覆盖范围大,通信成本与距离无关;可对所有地点提供相同的业务种类和服务质量(包括误码率和传输时延等);灵活性好(多种业务可在一个网内并存,对一个站来说支持的业务种类、分配的频带和服务质量等级等可动态调整);可扩容性好,扩容成本低,开辟一个新通信地点所需时间短;具有点对点通信能力;独立性好,是用户拥有的专用网,不像地面网中受电信部门制约;互操作性好,可使采用不同标准的用户跨越不同的地面网而在同一个 VSAT 网内进行通信;通信质量好(有较低的误码率和较短的网络响应时间);传播时延大。

VSAT 卫星通信网的特点是:面向用户而不是面向网络,VSAT 与用户设备直接通信而不经地面电信网络与用户设备进行通信;天线口径小,天线

口径一般小于 2.4 m,某些环境下可达到 0.5 m;智能化(包括操作智能化、接口智能化、支持业务智能化、信道管理智能化等)功能强,可无人操作;安装方便,只需简单的安装工具和一般的地基(如普通水泥地面、楼顶、墙壁等);低功率的发射机,一般功率为几瓦;集成化程度高,VSAT 从外表看只能分为天线、室内单元(IDU)和室外单元(ODU)三个部分;VSAT 站很多,但各站的业务量较小。一般用作专用网而不用作公用通信网。

按照 VSAT 的性质、用途、网络结构和某些特征可以进行不同的分类。按照 VSAT 支持的主要业务类型不同,可分为以话音业务为主的 VSAT 系统、以数据业务为主的 VSAT 系统和以综合业务为主的 VSAT 系统。从 VSAT 网采用的网络结构来看,可分为星状结构的 VSAT 系统和网状结构的 VSAT 系统。根据不同的安装方式可分为固定式、墙挂式、可搬移式、背负式、手提式、车载式、机载式及船载式等。按业务性质可分为固定业务的 VSAT 和移动业务的 VSAT 两种。按收发方式可分为单收站和双向站。

除了个别宽带业务外,VSAT 卫星通信网几乎可支持所有现有业务,包括话音、数据、传真、局域网(LAN)互连、会议电话、可视电话、低速图像、可视电话会议及数字音乐等。VSAT 网络可对各种业务分别采用广播(点→多点)、收集(多点→点)、点一点双向交互、点一多点双向交互等多种传递方式。

VSAT 的关键技术主要包括了下面几个方面:

①地球站技术 目标是 VSAT 的小型化,减低 VSAT 的成本。涉及到的技术有信号处理技术、集成电路技术、天线技术、接口技术、抗干扰技术、数字化技术、体系设计技术、标准化技术及加密技术等。②空间技术 目标是降低卫星的成本,在不增加卫星重量的基础上增加卫星的容量、提高卫星的利用率,从而加强卫星的竞争力。涉及到的技术有星上处理与交换技术、多波束/跳波束天线技术、星上信令能力、中低轨道卫星技术、增大卫星功率技术、宽带传输技术、干扰抑制技术和多信道 Modem 技术等。③网络技术 目标是将地球站和卫星结合起来,使 VSAT 达到最佳的性能价格比。涉及到的技术有多址访问技术、信道分配技术、网络控制技术、网络协议技术和网络安全技术等。

参考文献

1. 张更新. VSAT 卫星通信. 第一讲 VSAT: 卫

星通信的一个热门. 电信科学,1996,12(7)

2. G. MARAL. VSAT 网络. 北京: 电子工业出版社,1997 (史美林 英春)

shengwu jisuan

生物计算 (biocomputing) 利用生物系统固有的信息处理机理而研究开发的一种新的计算模式。生物计算研究包括器件和系统两个方面。利用有机(或生物)材料在分子尺度内构成有序体系,提供通过分子层次上的物理化学过程完成信息检测、处理、传输和存储的基本单元,称为**分子器件**。1974 年,A. L. Aviram 和 M. A. Ratner 首次提出分子整流器模型。1978 年,F. Carter 明确提出分子器件概念。尽管分子器件研究仍处于实验室阶段,但研究结果表明它的尺寸可望超越超大规模集成电路工艺的极限,而且它本身具有较高的自适应性、丰富的时变特性以及有利于大规模互连等优点。

生物计算系统不仅对器件有新的要求,其结构和计算原理也不同于传统的计算系统。它的结构一般是并行分布式的。信息的存储往往是短时记忆(快过程)和长时记忆(慢过程)的结合,是通过学习来完成的。它的计算则表现为复杂的动态过程,不存在精确的时间同步,甚至要在分维时间尺度上才能描述。M. Conrad 提出了进化学习方法和基于分子形状匹配的自组织机制。在分子器件方面,材料的选择及分子有序组装技术是关键。生物计算系统原理、信息编码及传递机制等都是重要的研究课题。值得注意的是利用细胞核中的脱氧核糖核酸(DNA)来完成计算。1994 年,美国南加州大学提出研制 DNA 计算机的设想,并用 DNA 计算方式求解了 7 个城市旅行推销员问题,显示了 DNA 计算的巨大潜力。合成具有特定序列的 DNA 分子,通过与酶的相互作用,就会发生反应。反应前的编码分子序列为所求解的问题,反应后的编码分子序列即为问题的解。DNA 分子可同时进行大量的(例如 10^{20})生化反应,所以 DNA 计算将是超大规模并行的。DNA 计算还有功耗极低和存储密度非常大的优点。但也有很多问题需要解决,包括生化反应慢,DNA 分子容易水解,操作有随机性和 DNA 分子之间难以通信等。

生物计算还处在发展的早期阶段,如果能研制出有实用价值的生物计算系统,可有效解决传统计算机难以解决的一些问题。一种较有希望的实现途径是通过硅基与碳基相结合的技术,实现半导体芯

片与生物芯片的系统集成。

参考文献

1. Conrad M ed. Molecular Computing. Special Issue of Computer, 1992, 25(11): 6~20

2. 韦钰, 甘强. 分子计算理论与神经网络. 东南大学学报, 1990, 20(6): 109~114 (甘强)

shengwu tezheng shibie

生物特征识别 (biometrics) 由计算机利用人体所固有的生理特征或行为特征来进行个人身份辨认和(或)验证的过程、方法与技术。生理特征与生俱来, 多为先天性的, 比如指纹、脸像、虹膜等; 而行为特征则是习惯使然, 多为后天性的, 比如语音、唇动等。

常见的生物特征识别技术主要有**指纹识别、人脸识别、虹膜识别、说话人识别、唇动识别**等。由于单个生物特征识别在鲁棒性和稳定性方面还不够完备, 因此出现了将多个特征综合考虑的**多模态生物特征融合技术**。

生物特征识别技术还是和谐人机交互的重要组成部分。在和谐的人机交互环境下, 计算机主动观察用户, 根据用户的行为或动作趋向推断其意图, 主动为其提供服务。生物特征识别技术将为计算机在非接触式的环境下主动识别用户的身份奠定坚实的基础。

发展历史

利用生物特征鉴别身份的技术历史悠久, 据考证, 早在公元前 7000 年—前 6000 年, 指纹即已经作为身份鉴别的方法在中国和古叙利亚有所应用, 比如留有匠人指纹的黏土陶器, 印有起草者指纹的文件等。1880 年, 科学家发现指纹的惟一性, 使得指纹用来作为身份识别的基础得以确认。19 世纪 90 年代, 人类学家 Alphonse Bertillon 将生物特征作为一门辨认的科学加以研究, 并用来辨识罪犯的身份, 后人将其称之为“Bertillonage”(贝迪永式人体测定法)。20 世纪 60 年代, Bell 实验室的 S. Pruzansky

提出了基于模式匹配和概率统计方差分析的说话人识别方法, 形成了说话人自动身份识别研究的高潮。1936 年, 眼科专家 Frank Burch 指出虹膜具有独特的信息, 可以用于身份识别。1987 年, Aran Safir 和 Leonard Flom 提出了利用虹膜图像进行自动识别的概率, 并获得专利。自 20 世纪 60 年代以来, 由于计算机及相关其他技术的发展, 基于生物特征的自动身份识别技术获得了迅猛的发展。

目前, 生物特征识别已经进入了实际应用阶段。国外许多高技术公司正在试图用虹膜、指纹、脸像、语音等特征取代人们手中的信用卡或密码, 并且已经开始在机场、银行、电子商务以及各种电子器具上进行了实际应用。国内也有多家研究机构和开发单位正在从事多种生物特征识别技术的研究与开发, 指纹识别、脸像识别等技术已经趋向成熟, 并逐渐走进人们的工作和生活中。

原理与特点

传统的身份识别方法, 通过识别一些标识个人身份的事物来进行身份的辨认。大体上可以分成两类: ①特定的持有物, 如身份证、信用卡、钥匙等; ②特定的知识, 如用户名和密码等。然而这些传统的方法存在容易丢失或被伪造、遗忘或记错等缺点, 而且这些方法无法区分真正的拥有者和取得身份标识物的冒充者, 一旦他人获得了身份标识物, 将可以拥有相同的权力。

与上述传统的身份识别方法相比较, 基于生物特征的身份识别技术由于使用了人体所固有的生理或行为特征, 因而具有: ①不易遗忘或丢失; ②防伪性能好, 不易伪造或被盗等优点。当然生物特征识别技术也存在着如下一些缺点: ①成本较高; ②存在拒识和误识现象。特别是后者是目前生物特征识别所面临的主要难点。如何降低拒识率和误识率, 提高识别的性能, 是目前研究的目标和重点。

各种生物特征识别技术的原理大致相同, 不同的只是一些具体的处理方法。图 1 为生物特征识别的基本原理框图。

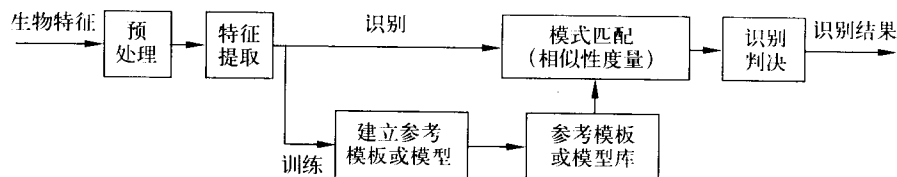


图 1 生物特征识别的基本原理框图

生物特征识别系统的实现过程包括训练和识别两个阶段。无论是训练还是识别,都需要首先对原始的生物特征信号进行预处理,滤除掉原始信号的不重要的信息及背景噪声等。然后进行特征提取,提取出反映信号特征的关键特征参数,以降低维数并便于后续处理。在训练阶段,每个用户输入若干次某种生物特征,系统经过上述预处理和特征提取后对其进行分析,据此建立每个用户的模板或模型库,或者对已在库中的该用户的模板或模型作适应性修正。由于该阶段为系统的每个用户都注册了自己的信息,所以又称之为注册阶段。在识别阶段,将待识别的用户所输入的生物特征的特征参数与在训练过程中建立的参考模板或模型加以比较,并经过一定的相似性准则进行识别判决。

分类

生物特征识别包括辨认和确认两大范畴。

辨认 试图识别出用户的身份,是一个多选一(1:N)的问题。系统输入某种生物特征,然后据此从库中选出与该特征相似性最优的模板或模型,并将其相似性和某个域值加以比较,若其大于该域值,则给出对应的用户的身份,否则,作出用户没有注册(不属于当前参考模板或模型库)的判决。可以看出辨认的过程,系统决策的选择数目为库中所包含的参考模板或模型的数目,因而辨认的性能随着库的规模增大而降低。

确认 试图确定某个用户是否具有其所声称的身份,是一对一(1:1)的问题。系统接受生物特征输入的同时声称该特征所对应的身份,然后系统将该特征与库中和所声称的身份对应的模板或模型进行匹配,并和某个相似性域值加以比较,给出接受(得到确认)或拒绝(拒绝确认)的判决,因此确认的性能与库的规模无关。

性能评价

生物特征识别除基本的正确率、速度、存储容量等性能评价的标准外,还有两个重要的统计性能指标:错误拒绝率(FRR)和错误接受率(FAR)。错误拒绝是指生物特征的拥有者被系统拒绝接受;而错误接受是指将冒充者识别为真正的生物特征的拥有者。上述两个错误率考虑了识别系统的整体性能,如果单纯从算法加以考虑,主要的指标为错误非匹配率(FNMR)和错误匹配率(FMR)。

假设拥有者和冒充者的特征参数符合一定条件的分布,则系统判决以及错误拒绝率和错误接受率的示意如图2所示。

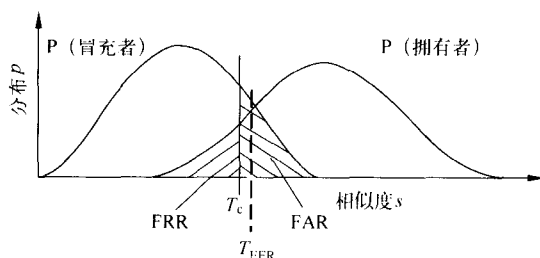


图2 拥有者和冒充者分布以及分类错误率

对于理想的系统来说,错误拒绝率(FRR)和错误接受率(FAR)都应该是零,即拥有者和冒充者的特征分布可以无错误地加以区分。而实际系统中两个错误率是相关的,而且和所给定的用于识别判决的相似性域值 T_c 有关。域值越大,系统接受的相似性要求也就越大,因而更容易造成错误拒绝,即错误拒绝率 FRR 越大,而错误接受率 FAR 则越小。反之亦然。系统往往需要在两个错误率之间取一个折中。用 ROC 曲线可以很好地反映两个错误率之间的关系,如图3所示。曲线上的点表示在某个给定的相似性域值下得到的错误拒绝率和错误接受率。图3中所标出的等错误率点(EER)表示错误拒绝率 FRR 和错误接受率 FAR 相等的点,此时的相似性域值为 T_{EER} ,在图2中亦有表示。有时,等错误率 EER 也可以作为指纹识别系统(参见指纹识别)的性能标准。

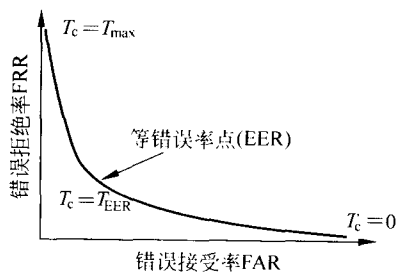


图3 ROC 曲线

发展趋势

目前,单个生物特征识别技术中,还没有哪一种能做到完美无缺,它们在鲁棒性、稳定性等方面均存在一定的问题。比如在噪声环境中利用语音的说话人识别系统将不能很好地运行;对于双胞胎,仅仅使用脸像特征也不能很好地区分;而对于指纹识别技术,有5%左右的人不能得到很好的指纹特征等。

将多种生物特征进行融合,综合考虑其总体效果可以有效地提高系统的性能,取得更高的鲁棒性和稳定性。如脸像、语音、指纹等生物特征的融合系

统,当在噪声环境中语音不能发挥很好作用时,其他两个生物特征仍然能够得到很好的辨识效果。这正是多模态生物特征融合所研究的内容(参见多模态生物特征融合),也是今后生物特征识别领域的发展趋势。

参考文献

1. <http://www.biometrics.org>
2. Nalini K Ratha, Andrew Senior, Ruud M Bolle. Automated Biometrics. In: Proceedings of ICA-PR-2001, Rio de Janiero, Brazil, 2001
3. Jain A, Hong L, and Pankanti S. Biometrics: Promising Frontiers for Emerging Identification Market. Comm. ACM, 91~98, February, 2000

(吴志勇 蔡莲红)

shidian yansuan

时段演算 (duration calculus) 一种实时区间时态逻辑。它将布尔函数在区间上的积分进行形式化,从而用来描述和推导离散状态系统的实时和逻辑特性。

时段演算的研究始于1989年。当时Esprit的研究项目ProCoS正寻求设计严格安全系统的形式技术,应用项目的需要推动了时段演算的研究。该演算是由周巢尘, C. A. R. Hoare 和 A. P. Raun 所提出。

时段演算已应用于若干实例,如煤气燃烧器、铁路岔口控制、水位控制、自动导航、Occam语言的实时语义、描述调度程序的实时行为和电路设计等方面。

ProCoS项目的一个研究实例就是要对如下煤气燃烧器需求进行形式化:“如果对系统观察的时间大于60s时,那么漏气的时间占整个时间的比例应小于1/20。”应用数学分析可直接地对这个需求进行形式化,结果是

$$(e - b) \geq 60 \text{ s} \Rightarrow 20 \int_b^e \text{Leak}(t) dt \leq (e - b)$$

这里Leak是一个布尔函数,它表示煤气燃烧漏气状态。函数是从实数R(表示时间)到{0,1}的函数,其中1表示系统正处于该状态,0表示系统不处于该状态。观察的区间采用闭区间,并且用b表示开始和用e表示结束。积分被认为是从状态函数和区间到实数的函数: $\int: S \rightarrow (I \rightarrow R)$ 。其中S表示状态集(即布尔函数),I表示闭区间集。因此区间时态逻辑(这里被扩展成连续时间模型)被作为它的基础逻辑,并且区间函数 $\int S, \int P \dots$ 变成了演算的区

间变量。这里S和P是状态。如此有 $\int 1 = e - b$,并且我们用l作为它的缩写,即是区间的长度。因此上面的需求可以更简单地描述为:

$$\text{Req: } l \geq 60 \Rightarrow 20 \int \text{Leak} \leq l$$

通过用积分定义一个 $\lceil \cdot \rceil$ 的运算,我们可以表示一个状态在区间上持续的出现: $\lceil S \rceil \triangleq (\int S = l \wedge l > 0)$, $\lceil S \rceil$ 在一个区间上成立要求这个区间为非点区间,并且状态S在区间上(几乎)处处取值为1。如此下面的公式

$$\text{Dec1: } (\lceil \text{Leak} \rceil \Rightarrow l \leq 1)$$

形成了煤气燃烧器可行的设计决策,即一次失败的点火在1s内是可以被检测到而且能够被停止的。

区间时态逻辑的模式词是“切变”(;) ,语义上定义为

$$A; B[b, e] \triangleq \exists m: b \leq m \leq e. A[b, m] \wedge B[m, e]$$

它的含义是公式A;B在区间[b,e]上成立,当且仅当区间可以被切成两个子区间,并且使得A在第一个区间上成立,B在第二个区间上成立。切变是个连接运算,它使设计者能够通过组装几个子区间上的性质,从而刻画整个系统在观察区间上的性质。例如利用切变,可以对煤气燃烧器另外一个设计决策进行形式化。它要求系统在两次漏气之间至少要等待30秒,从而使我们排除了煤气燃烧器频繁的点火故障。

$$\text{Dec2: } (\lceil \text{Leak} \rceil; \lceil \neg \text{Leak} \rceil; \lceil \text{Leak} \rceil \Rightarrow l \geq 30)$$

通常的模式词可以由切变来定义: $\Diamond A \triangleq \text{true}; A; \text{true}$ 和 $\Box A \triangleq \neg \Diamond \neg A$ 。 $\Diamond A$ 在一个区间上为真是指A在某一个区间上为真; $\Box A$ 在一个区间上为真是指A在每一个区间上都为真。

假设状态具有有限变化性。我们可以得到6个公理。它们类似测度论中关于有限区间集合的测度公理,并且构成了一个相对完全的演算,即时段演算。它们是

1. $\int 0 = 0$
2. $\int P \geq 0$
3. $\int P + \int Q = \int (P \vee Q) + \int (P \wedge Q)$
4. $\left(\int P = r + s \right) \Leftrightarrow \left(\int P = r \right); \left(\int P = s \right)$
($r, s \geq 0$)
5. $(\lceil \neg \vee \lceil P \rceil; \text{true} \vee \lceil \neg P \rceil; \text{true})$
6. $(\lceil \neg \vee \text{true}; \lceil P \rceil \vee \text{true}; \lceil \neg P \rceil)$

其中 $\neg \Delta (l=0)$ 。公理 5 和 6 刻画了状态 P 的有限变化性。也就是,对于任何非点区间都可以分成有限次 P 和 $\neg P$ 的交换。

应用该演算,可证明下面的公式是该演算的定理,从而证明了上面两个设计决策的确满足需求的正确实现。

$\square Dec1 \wedge \square Dec2 \Rightarrow Req.$

在实时系统形式化领域的研究中,时段演算被公认是一种成功的形式化方法。目前时段演算的研究还在不断地发展完善之中。

参考文献

1. Zhou Chaochen. A Calculus of Durations. Information Processing Letters, 1991, 40(5)
2. 李晓山,周巢尘.时段演算综述.计算机学报,1994,17(11),842~851 (李晓山 周巢尘)

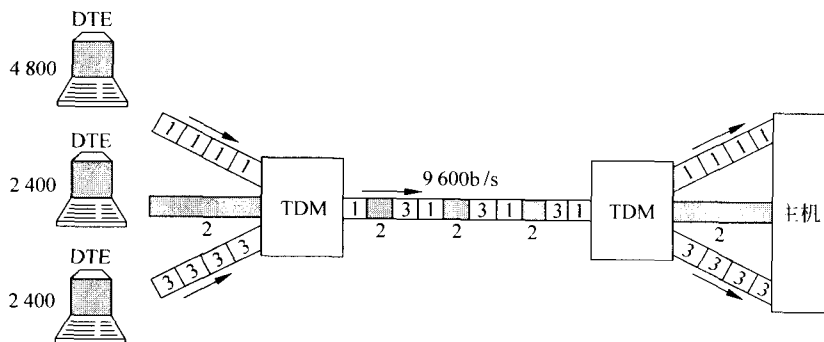


图1 时分多路复用技术

TDM 可以设计成按位、按字节、按字符、按字或按任何其他多位方式来对每个终端装置进行扫描。主要扫描技术有:

(1) 隔位扫描技术 隔位扫描多路复用器对来自每个数字输入信号的一位进行取样,然后按其被取样的次序串行发送每一位。这种交错的位图称为位流。

在位流的前后都加上同步位(附加位)。同步位和同步位之间的数据形成一帧。

无论每个信源是否有数据要传送,都发送信源的时隙。如果有一个信源空闲,多路复用器便在这个空时隙中插入一个零字符,而不管它原来发送的是“1”,“0”或控制字符。隔位扫描技术因有空时隙(如果有一个以上的输入端不发送数据的话)而使传输容量有所浪费,但这是为了保证维持原有的取

shifen duolu fuyong

时分多路复用 (time division multiplexing, TDM) 采用分时技术把传输线路的可用时间分成时隙,并按一定规律将时隙分配给多个输入信号的多路复用技术。

时分多路复用原理如图 1 所示,其工作过程为:

- (1) 接到多路复用器的每个终端都分配到一个时隙。
- (2) 多路复用器对每个终端装置进行扫描,以确定它是否有字符要传送。
- (3) 如果终端装置没有字符要传送,多路复用器便发送一个空(零)字符,以保持序列顺序。
- (4) 因此,图中三个终端装置的数据速率之和就等于多路传输线路的数据速率 $2\ 400\text{ b/s} + 2\ 400\text{ b/s} + 4\ 800\text{ b/s} = 9\ 600\text{ b/s}$ 。

样位序列所作出的牺牲。

隔位扫描技术的数字数据传输效率约为 90%~98%。

(2) 隔字符扫描技术 隔字符扫描技术是对来自每个数字输入信号的一个字符(而不只是一位)进行取样,然后像隔位扫描技术一样,发送出一个复合的数据流。①字符可以由任何给定数目的位组成;这里以采用 8 位字符为例。②与隔位扫描 TDM 相似,同步字符置于数据流的前后,以提供必不可少的定时。③隔字符扫描多路复用器接收每个数字输入信号的开头 8 位,并按接收它们的次序以串行方式输出。

数字信号分级

按多路传输线的数据速率对数字信号分级如下:

(1) 话音信号通常是用脉码调制来编码的。为了用数字方式来充分代表一个模拟话音信号,对模拟话音信号取样的频率必须至少为 8 000 次每秒。一个 8 位字可用代表每个取样,所以,话音信号数字化的结果便是一个 $8\,000 \times 8$ (位字)的数据流,数据速率为 64 000 b/s。这一信号被定义为 DS0 级数字化话音信号,它由 8 000 个 8 位字组成。这是数字多路复用分级中的第一级。

(2) DS1: DS1 信号由在一条线路上多路传输的 24 个 DS0 信号组成。这一信号包含 24 个数字化话路。

(3) DS1C: DS1C 信号是使两个 DS1 信号的输出共同在一条线路上多路传输而形成的,它包含 48 个数字化话路。

(4) DS2: DS2 信号是使两个 DS1C 信号的输出共同在一条线路上多路传输而形成的。它包含 96 个数字化话路。

(5) DS3: DS3 信号是使 7 个 DS2 信号的输出共同在一条线路上多路传输而形成的。它包含 672 个数字化话路。

(6) DS4: DS4 信号是使 6 个 DS3 信号的输出

共同在一条线路上多路传输而形成的。它包含 4 032 个数字化话路。

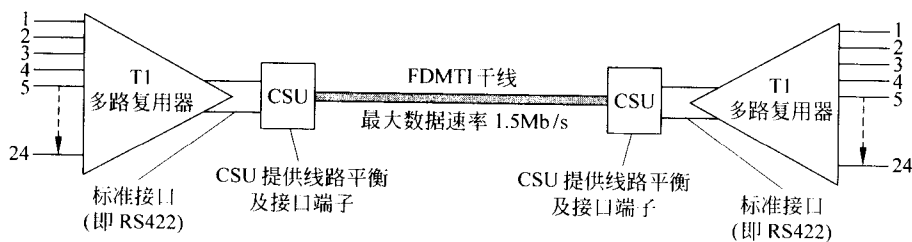
数字信号分级见表 1。

表 1 TDM 数字信号分级

级别	比特率	话路	传输系统
DS0	64 kb/s	1	8 位 \times 8 000 次取样
DS1	1.544 Mb/s	24	双绞电缆-T1 无线电中继系统
DS1C	3.152 Mb/s	48	双绞电缆 T1C-T1D
DS2	6.132 Mb/s	96	本地用电缆-T2 无线电中继系统-2GHz
DS3	44.736 Mb/s	768	无线电中继系统-6GHz, 11 GHz 光波系统-FT3
DS4	274.176 Mb/s	4 032	无线电中继系统-18 GHz 光波系统-开发中 同轴电缆系统-T4M

T1 多路复用器

在数字信号分级中,DS1 信号(包含多路传输的 24 个数字化话路)是由叫做 T1 的多路复用器产生的(见图 2 所示)。



* 输入信号可以是原始的数字数据或数字化的话音信号。

图 2 T1 多路复用器

T1 多路复用器常常脱离数字信号分级而单独使用,数据或数字化话音的组合均可通过 T1 来多路传输,因为 T1 对数据和数字化话音是不加区别的。

T1 多路复用器是一个真正的时分多路复用器,因为它可以处理同步输入、异步输入、数据输入或数字化话路输入。

几乎所有的 T1 多路复用器不是隔位扫描多路复用器就是隔字符扫描多路复用器,均可在全双工点对点线路上工作。

T1 多路复用器必须通过一个信道服务部件(CSU)与线路连接。信道服务部件提供线路平衡和接口端子。接口通常是 RS422 标准接口。

同步光纤网—同步数字体系(SONET/SDH)

同步光纤网络(SONET)是在光纤电缆上同步传输数字信号的网络传输技术(参见同步光纤网)。这种传输技术通过集合多信道产生分级结构来提供几个不同性能的级别。国际电信联合会(ITU)规定的各级数据速率如表 2 所示。它提供了许多高级特性,包括 51.84 Mb/s ~ 9.953 Gb/s 的传输速率、高效传输、低成本、优秀的管理资源和处理任意数据的能力,其向后兼容性以及带冗余和自修复特点的拓扑结构等。国际电信联合会将这种标准定名为同步数字体系(SDH)。

表2 国际电信联合会 (ITU) 规定的速率

SONET	ITU	数据速率 (Mb/s)
STS-1/OC-1		51.84
STS-3/OC-3	STM-1	155.52
STS-9/OC-9	STM-3	466.56
STS-12/OC-12	STM-4	622.08
STS-18/OC-18	STM-6	933.12
STS-24/OC-24	STM-8	1 244.16
STS-36/OC-36	STM-12	1 866.24
STS-48/OC-48	STM-16	2 488.32

统计时分多路复用

统计时分多路复用 (STDM) 是一种效率更高的 TDM 方法。它只适用于真正的数字数据 (如计算机的输出), 不适用于经过转换的模拟数据 (如 PCM 编码的语音信号)。

在 TDM 中, 每个输入端不管其状态如何 (空闲或发送), 都分配有一个时隙, 如图 3 (a) 所示。STDM 与 TDM 的不同之处在于它只把时隙分配给要发送数据的输入端, 不分配给空闲的输入端, 如图 3 (b) 所示。这就能节省传输线的时间和空间, 从而提高效率。

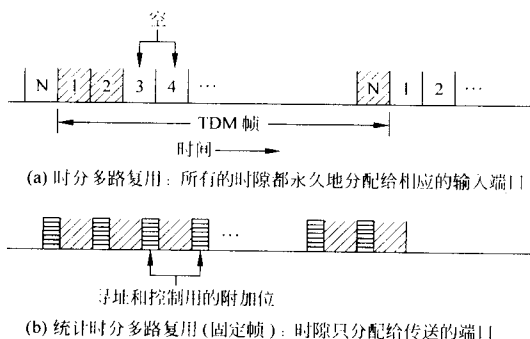


图3 TDM 与 STDM 的比较

参考文献

1. 胡道元. 计算机局域网 (第三版). 北京: 清华大学出版社, 2002
2. Sheldon T. LAN Times Encyclopedia of Networking. McGraw-Hill Inc., 1994 (胡道元)

shijian fuzaxing

时间复杂性 (time complexity) 用时间来度量的计算复杂性, 它是最重要的复杂性度量。粗略地说, 算法的时间复杂性就是该算法的执行时间。

算法 C 对每一个可能的输入长度 x , 若执行时间为 $t(x, C)$, 则 $W_i(n, C) = \max \{t(x, C) | x \text{ 的长度为 } n, \text{ 即 } |x| = n\}$ 给出用该算法解输入规模为 n 的问题所需要的最长时间。称这种时间复杂性为最坏情况时间复杂性, 如果把 $D_n = \{x | |x| = n\}$ 看成一个概率空间, 出现输入 x 的概率等于 $p(x)$, 于是算法解输入规模为 n 的问题所需要的平均计算时间为 $A_i(n, C) = \sum_{x \in D_n} p(x) t(n, C)$ 。计算时间与执行算法的计算机或计算模型有很大的关系。如在图灵机模型中, 计算时间是指图灵机停机前执行的总步数。但从理论上讲, 根据相似性和对偶性原理: 同一算法在各种计算模型上计算时间复杂性并无很大的差别。在实用上, 常用基本操作数作为时间的度量, 例如对排序问题, 以被排序元素的比较次数为基本操作数, 又如矩阵乘法以矩阵元素的相乘次数为基本操作度量, 这样计算时间的目的是把影响算法执行时间的次要因素舍弃, 而着重计算影响算法执行时间的主要因素——基本操作数, 从而简化了时间复杂性的计算但又不损害对该算法性能的评价。

时间复杂性函数在 $\bigcup_k O(n^k)$ 中的算法称为多项式时间算法, 超过此范围的, 统称为指数时间算法。对于下面 5 个算法:

算法	时间复杂性
A_1	n
A_2	$n \log n$
A_3	n^2
A_4	n^3
A_5	2^n

表 1、表 2 分别给出特定时间内它们能解决问题的规模和计算机速度增加 10 倍后它们计算能力的增强情况。

表1 复杂性限制的问题大小

算法	时间复杂性	最大问题规模		
		1s	1min	1h
A_1	n	1 000	6×10^4	3.6×10^6
A_2	$n \log n$	140	4 893	2.0×10^5
A_3	n^2	31	244	1 897
A_4	n^3	10	39	153
A_5	2^n	9	15	21

表2 计算机速度提高10倍的效果

算法	时间复杂性	加速前最大问题规模	加速后最大问题规模
A_1	n	s_1	$10s_1$
A_2	$n \log n$	s_2	$4.6s_2$
A_3	n^2	s_3	$3.16s_3$
A_4	n^3	s_4	$2.15s_4$
A_5	2^n	s_5	$s_5 + 3.3$

从上表可以看出,对多项式时间算法,计算机计算能力的提高对问题的可解规模明显增加,而对指数时间算法,计算机计算能力的提高使问题的可解规模几乎不增加。因此计算机科学家公认简单区分算法效率高低的分界线是它为多项式时间算法还是指数时间算法,凡一个问题有多项式时间算法则称该问题是易解的,称它属于P类。反之,如一个问题还没有多项式时间算法则称该问题是难解的。寻找问题的多项式时间算法或者证明该问题是难解的一直是算法研究的最重要的内容。如果问题有不确定性的多项式时间算法,则称它属于NP类。NP类中最难解的问题称为NPC类。P类是否就等于NP类是当代理论计算机科学中最著名而重要的未决难题。

随着计算机系统向并行、并发、分布式和巨型体系发展,遂有并行、分布式算法的兴起,并行算法的时间复杂性和空间复杂性有相互折算的关系,空间耗费大,即并行程度高,则并行算法的执行时间可以降低。反之,算法的并行程度低,则并行算法的执行时间就会增加。设计并行时间复杂性态好的并行算法和分析并行算法的时空折算关系以及一个问题的并行算法时间复杂度下界是并行算法研究主要关心的问题。

参考文献

1. 朱洪,段振华,陈增武,周克成. 算法设计和分析. 上海: 上海科学技术文献出版社, 1989
2. Cormen T H, Leiserson C E, Rivest R L. Introduction to Algorithms. The MIT Press, 1990
3. Akl S G. The Design and Analysis of Parallel Algorithms. N. J.: Prentice Hall (朱洪)

shitai luoji

时态逻辑 (temporal logic) 关于随着时间变化而不断改变其值的动态变元(称为时序变元)的一种模态逻辑。它除含有经典逻辑的逻辑联结词和量词外,还含有一些时态算子。

时态逻辑包括命题时态逻辑和谓词时态逻辑,选择不同的时态算子将导致不同的时态逻辑系统。在计算机科学中应用较为广泛的命题线性时态逻辑系统 PLTL,是由 A. Pnueli 和 Z. Manna 给出的。PLTL 包含可数无穷多个命题变元,以及逻辑联结词 \neg (否定)、 \wedge (合取)、 \vee (析取)、 \supset (蕴含)与 \equiv (等价)和时态算子 \square (意为“任一时刻”)、 \diamond (意为“某一时刻”)、 \bigcirc (意为“下一时刻”)与 \mathcal{U} (意为“直到”)。PLTL 的合式公式可归纳定义如下:

- (1) 命题变元 P 是合式公式;
- (2) 若 w, w_1 和 w_2 是合式公式,则 $(\neg w), (w_1 \wedge w_2), (w_1 \vee w_2), (w_1 \supset w_2)$ 和 $(w_1 \equiv w_2)$ 均为合式公式;
- (3) 若 w, w_1 和 w_2 是合式公式,则 $(\square w), (\diamond w), (\bigcirc w)$ 和 $(w_1 \mathcal{U} w_2)$ 均为合式公式;
- (4) 每个合式公式皆可通过有限次应用(1), (2), (3)得到。

PLTL 包含以下 10 条公理和 3 条推理规则:

公理 1: $\neg \diamond w \equiv \square \neg w$

公理 2: $\square(w_1 \supset w_2) \supset (\square w_1 \supset \square w_2)$

公理 3: $\square w \supset w$

公理 4: $\bigcirc \neg w \equiv \neg \bigcirc w$

公理 5: $\bigcirc(w_1 \supset w_2) \supset (\bigcirc w_1 \supset \bigcirc w_2)$

公理 6: $\square w \supset \bigcirc w$

公理 7: $\square w \supset \bigcirc \square w$

公理 8: $\square(w \supset \bigcirc w) \supset (w \supset \square w)$

公理 9: $(w_1 \mathcal{U} w_2) \equiv (w_2 \vee (w_1 \wedge \bigcirc(w_1 \mathcal{U} w_2)))$

公理 10: $(w_1 \mathcal{U} w_2) \supset \diamond w_2$

命题重言规则: 若 u 是命题重言式,则 $\vdash u$

假言推理规则: 若 $\vdash u \supset v$ 且 $\vdash u$,则 $\vdash v$

\square 引入规则: 若 $\vdash u$,则 $\vdash \square u$

应用上述公理和推理规则经有穷步可推导出来的合式公式称为该系统的定理。

在时态逻辑中,时间的结构可以是线性的或分支的,离散或连续的,基于时间点的或时区的。基于不同的应用背景,可采用不同的时间结构。PLTL 采用线性的、离散的且与自然数同构的时间结构。PLTL 的语义解释是一个无穷状态序列 $\sigma = s_0, s_1, s_2, \dots$, 其中每个 s_i 是对命题变元的一个赋值。若令 $\sigma^{(i)} = s_i, s_{i+1}, \dots$, 且用 $\sigma \models w$ 表示时态公式 w 在解释 σ 下为真,则时态算子的含义定义如下:

$\sigma \models \square w$ 当且仅当 对任意 $i \geq 0$, 均有 $\sigma^{(i)} \models w$

$\sigma \models \diamond w$ 当且仅当 存在 $i \geq 0$, 使 $\sigma^{(i)} \models w$

$\sigma \models \bigcirc w$ 当且仅当 $\sigma^{(1)} \models w$

$\sigma \models w_1 \mathbin{\%} w_2$ 当且仅当 存在 $i \geq 0$, 使 $\sigma^{(i)} \models w_2$ 且对任意 $j (0 \leq j < i)$ 均有 $\sigma^{(j)} \models w_1$

时态逻辑的发展是与程序规约和程序验证紧密相关的。程序的行为是一种动态现象,其状态随着时间的推移不断改变,且可能不断影响外部环境。持续不终止的并发反应式程序的动态行为是经典逻辑和霍尔逻辑所不能描述的。为此,R. Burstall 在 1974 年首先建议使用模态逻辑进行程序推理,A. Pnueli 和 Z. Manna 建立了可用于程序规约和验证的时态逻辑系统。

时态逻辑具有很强的表达能力,可表达程序的安全性(如部分正确性、互斥性和无死锁性等)、活性(如终止性、完全正确性和响应性等)和事件优先性,是一种研究并发程序尤其是持续不终止的反应式程序(如操作系统、网络通信协议等)的强有力的形式化工具。目前,时态逻辑已广泛应用于程序的规约、验证和形式化开发,以及程序自动综合和模块化规范合成等并发程序设计的几乎所有方面。通过引入全局时钟或在时态算子上增加时间约束,时态

逻辑亦可用于实时系统的规约和验证。

参考文献

Emerson E A. Temporal and Modal Logic. in: Handbook of Theoretical Computer Science. Elsevier Science Publishers, 1990 (李舟军)

shixu xitong

时序系统 (timing system) 在同步控制的计算机中,为执行指令提供的各种以时钟脉冲为基础的、表示时间先后次序的控制信号。为了保证计算机各个部件协调工作,这些信号必须有一定的顺序。

计算机的时序系统负责向计算机内的各部件提供定时信号,使各部件在规定的时间内完成规定的操作。完成一条机器指令的时间叫一个**指令周期**,一个指令周期分成若干个**机器周期**,每个机器周期又分成若干个节拍来执行若干个微操作,(通常节拍宽度与时钟周期宽度相同),每个节拍又设置若干工作脉冲,形成一个严格的时序系统。

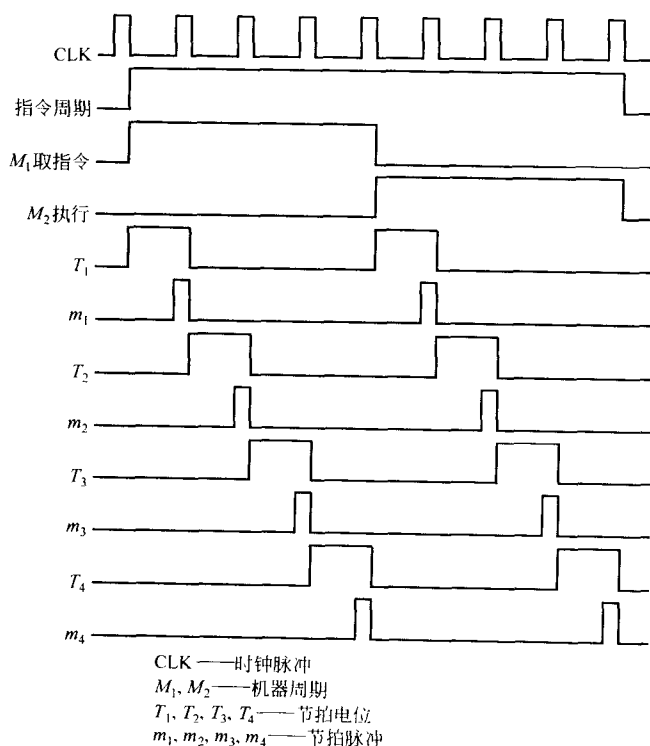


图1 时序信号波形图

时序系统举例:如果所执行的那条指令有两个机器周期:取指周期和执行周期,每个周期设置四个节拍,每个节拍设置一个工作脉冲,则其时序信号波形如图1所示。在取指周期,第一节拍 T_1 ,发送指令地址,其工作脉冲 m_1 把指令地址打入内存地址寄存器。第二节拍 T_2 发读内存命令。第三节拍 T_3 等待内存读出,第四节拍 T_4 读出指令,经数据总线送到指令寄存器,其工作脉冲 m_4 把指令打入指令寄存器中。

参考文献

金兰. 计算机组织与结构. 北京: 高等教育出版社, 1986
(谢树煌)

shishi huizhi

实时绘制 (real time rendering) 一类可以在任意视点条件下,至多在 $\frac{1}{24}$ s内完成整个场景的绘制并输出一帧图像的过程和技术。在 $\frac{1}{24}$ s内必须完成绘制过程并输出结果,将保证在用户看来这些生成的图像序列是连续变化的。实时绘制的要求是要在保证时间约束的前提下,尽可能提高所绘制图形的真实感。实时绘制技术主要用于交互式三维设计、三维计算机游戏、实时仿真器、虚拟现实系统等场合。

一般说来,场景是由一系列具有不同材质的造型元素组成的。普通的绘制算法的效率与景物空间复杂度相关,即生成一幅场景画面,需要遍历并处理场景中的每一个造型元素,至少具有 $O(N)$ 的计算复杂度(N 为场景中造型元素的个数)。然而,理论上,场景的复杂性是无限的,且任何硬件在单位时间内的处理能力总是有限的,所以具有 $O(N)$ 算法复杂度的绘制算法难以胜任实时绘制的任务。因此,实时绘制方面研究的关键在于通过研究场景复杂性和人类视觉感知规律等相关因素的内部联系,设计出与场景复杂性基本不相关或弱相关的绘制算法,即具有 $O(1)$ 或 $O(\log(N))$ 算法复杂度的绘制算法。

常见的实时绘制技术包括可见性剔除、层次细节、基于图像的绘制等。

(1) 可见性剔除 这类算法可以用较小的代价判断哪些造型元素一定不可见。当场景被组织成层次结构时,对任意观察参数,可见性剔除算法在绘制过程的较早阶段以较少的处理时间成批地淘汰掉不可见的造型元素,这样可使得绘制时间不随场景的

规模而线性增长。

(2) 层次细节 其基本原理是,利用透视投影的特性——距离当前观察视点越远的物体,在成像平面上的投影面积越小,那么远处的物体在绘制阶段可用较少的等效造型元素来表现它,这使得绘制时间大大缩短。

(3) 基于图像的绘制 这种方法首先在一系列视图上对整个场景进行预绘制,并将获得的图像经适当组织形成图像库。然后在绘制过程中,根据任意观察参数,从图像库中抽取出合适的图像像素或块,再经过适当处理后,来填充待绘制图像上的像素或区域。

参考文献

1. Akenine-Moller T and Haines E. Real-Time Rendering. 2nd edition. A. K. Peters Ltd., 2002
2. Cohen-Or and Chrysanthou Y and Silva C. A Survey of Visibility for Walkthrough Applications. In: Proc. of EUROGRAPHICS '00, Course Notes, 2000
3. Paul S Heckbert and Michael Garland. Survey of Polygonal Surface Simplification Algorithms. Course SIGGRAPH '97
(鲍虎军 华炜)

shiti lianxi moxing

实体联系模型 (entity-relationship model)

一种适合于应用需求描述的数据模型。又称E-R模型。实体联系模型的基本语义单位是实体和联系。实体代表现实世界中客观存在的事物,联系代表客观事物间的关系。实体联系模型是由Peter, P. S. Chen于1976年提出的,广泛适用于软件系统设计过程中的概念设计阶段。实体联系模型是通过实体型及其间的联系型来反映现实世界。实体联系模型可以形象地用图形表示,称为E-R图。

E-R图中,矩形框代表实体型;菱形框代表联系型;椭圆形框代表实体型和联系型的属性,相应的名字记入框中。联系型及其涉及的实体型之间以线段连接,并在线段的端部标注联系的种类(1:m, m:n或1:1)。

下面是表示某工厂物资管理概念模型的E-R图的例子。物资管理涉及的实体型有:仓库、零件、供应商、项目、职工等。这些实体型之间的联系如下:①仓库和零件间具有多对多($m:n$)的联系,即一个仓库中可以存放多种零件;一种零件可以存放在多个仓库中。②仓库和职工间具有一对多(1:m)的联系,即一个仓库可以有多个职工担任保

管员,而每个职工只能在一个仓库工作。③职工之间具有一对多(1:n)的联系,即仓库主任领导若干保管员。④供应商、项目和零件三者之间有多对多(m:n:p)的联系。

描述以上实体及联系型的 E-R 图如图 1 所示。

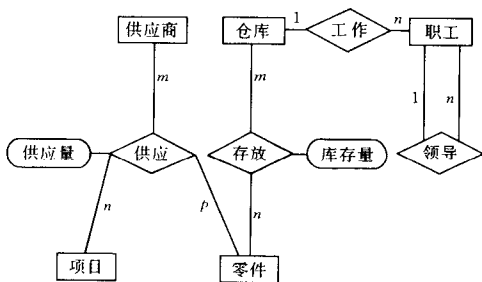


图 1 E-R 图

参考文献

Yao S B. Principles of Database Design. Vol. 1, Logical Organizations. Englewood Cliffs: Prentice - Hall.

(胡运发 周傲英)

shiti lianxitu

实体联系图 (entity relationship diagram)

用图形来描述概念模型的各实体间的联系的一种表示方法。又称 ER 图。现实世界的概念模型有很多表示方法,其中最常用的是 P. P. S. Chen 于 1976 年提出的实体联系模型(参见**实体联系模型**)。ER 图是实体联系模型的图形表示,常用于数据库设计中的概念建模,亦可用于其他软件设计的建模。ER 图提供了表示实体、属性、实体之间联系的方法。

实体:用矩形表示,矩形框内写明实体名。

属性:用椭圆形表示,并用无向边将其与相应的实体连接起来。

联系:用菱形表示,菱形框内写明联系名,并用无向边分别与有关实体连接起来,同时在无向边旁标上联系的类型。实体之间的联系有一对一、一对多、多对多的联系,分别用 1:1、1:n、m:n 来表示。

参考文献

萨师煊,王珊. 数据库系统概论(第三版). 北京:高等教育出版社,2000

(李宣东 王珊)

shiwu chuli

事务处理 (transaction processing) 数据库

管理系统中管理、协调事务以确保其正确执行的机制和过程。**事务元**(简称事务)是为完成一项特定业务而可单独执行的一组有序的动作,或该组动作的一次执行。数据库管理系统必须保证事务的执行具有 ACID 特性。其中,A 代表原子性,指的是事务的所有动作在数据库中要么全部执行要么全部不执行;C 代表一致性,指的是事务在独自运行(即没有其他事务并发执行)的情况下保持数据库的一致性,将数据库从一个一致状态转变为另一个一致状态;I 代表隔离性,指的是尽管系统中可以有多个事务在共同时间段内同时执行,即并发执行,但每一个事务都感觉不到系统中有其他事务在并发执行,在它看来,任何一个其他事务或者在本事务前执行,或者在本事务之后执行;D 代表持久性,指的是一个事务成功完成后,它对数据库的改变必须是永久的。

事务的原子性和持久性由数据库管理系统的恢复管理器来实现,保证当发生故障时一个事务对数据库的修改要么全部执行,要么全部不执行。恢复管理器主要采用日志、备份等技术。

单个事务的一致性由应用程序员负责,即应用程序员保证事务将数据库从一个一致状态转换到另一个一致状态。同时数据库管理系统对完整性约束的自动检查也提供了对事务一致性的支持。

事务的隔离性,即多个事务并发执行的正确性由数据库管理系统(DBMS)的并发控制机制提供支持,通过对并发事务的合理调度来保证每个事务的一致性。

当前商品化的关系数据库管理系统一般都能对事务处理提供很好的支持。

参考文献

1. Silberschatz A, Korth Henry F, Sudarshan S 著. 数据库系统概念. 杨冬青,唐世渭等译. 北京:机械工业出版社,2000

2. Gray J. Transaction Processing: Concepts and Techniques. Morgan Kaufmann Publishers, 1993

(杨冬青)

shiwu jincheng jiankongqi

事务进程监控器 (transaction process monitor)

客户与服务器之间数据传输的控制程序。事务进程监控器(TP 监控器)是一个软件中间件,它负责提供对各种资源管理器(如数据库管理系统、操作系统、用户界面、消息软件等)的访问,并且

为事务型软件的开发人员提供统一的接口。

事务进程监控器的功能主要包括:

(1) 调度 从客户端(终端)来的请求必须映射到实现应用服务的服务器程序;

(2) 服务器管理 负责启动各种服务器,负责负载均衡和所有相关的任务;

(3) 认证和授权 在执行请求服务前 TP 监控器必须了解请求,用户不是登录到操作系统,而是注册到事务进程监控器;

(4) 资源管理 负责管理终端、数据库、应用程序、用户和所有事务处理系统所涉及到的其他组件;

(5) 系统操作 TP 监控器必须向操作者提供足够的信息来调整系统,在普通操作过程中产生问题(如终端断开,服务器问题)时通知他们;

(6) 恢复 系统崩溃后,事务进程监控器负责重新建立事务处理环境。

事务进程监控器基本结构如图 1 所示。事务进程监控器不是简单地将消息传给应用服务器。当消息到达时,必须将它们放入到队列中,因此,需要一个队列管理器来管理到达的消息。此外,消息还可以存到稳定存储器中。这样,消息一旦收到,就会最终被执行,哪怕系统发生故障。事务进程监控器的进一步功能是权限控制和应用服务器管理(例如服务器启动和消息分发到各服务器)。事务进程监控器通常还提供日志、恢复和并发控制功能。这样,如果需要的话,应用服务器就能直接实现事务的不可再分割性、一致性、隔离性与持久性(ACID)。

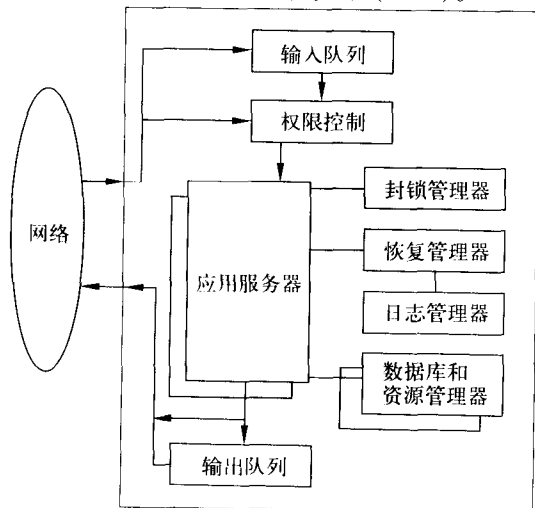


图1 事务进程监控器构成成分

事务进程监控器还为输出消息提供永久性队列服务。因此,服务器可以把消息输出作为事务的一部分。事务进程服务器保证,只要(而且仅当)事务提交了,消息就能发出去,这样,事务进程服务器不仅对数据库内部的动作,而且对数据库之外的消息传送都提供了 ACID 特性。

除了以上设施之外,许多事务进程监控器为终端这样的哑客户提供了表示功能,以帮助建立选单和表格界面。

参考文献

1. Gray J, Andreas R. Transaction Processing: Concepts and Techniques. San Francisco, Morgan Kaufmann, 1993
2. Silberschatz A, Korth Henry F, Sudarsham S. Database System Concepts (3rd edition). McGraw-Hill, 1999

(孟小峰)

shiwuyuan

事务元(transaction) 为完成一项特定业务而可单独执行的一组有序的动作,或该组动作的一次执行。事务元简称事务。例如,银行业务中的存款或取款、电子商务中的交易(包括验证、付款、结算等步骤)、各种票务中的订票或退票、电子设备间的数据交换过程(包括建立连接、交换数据、断开连接等步骤)、对数据库的一组有序操作(可以是增加、删除、修改、查询和存储等操作的各种组合)等都是事务元。从软件实现的角度看,一个事务元既可指为完成一项特定任务而可单独运行的一段程序,也可指该程序的一次运行,在运行过程中它一般都要与相关数据库、信息库乃至知识库进行一次或多次交互。

(何新贵)

shijue jisuan lilun

视觉计算理论(theory of vision computing)

通过建立人类视觉功能的数学模型及其解的算法实现来阐明视觉具有可计算性的原理。

视觉计算理论的奠基者 D. Marr 对视觉的可计算性进行了论证,并提出以图 1 所示的计算流程来予以说明。

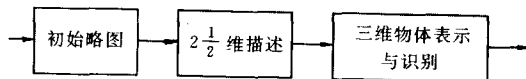


图1 说明视觉计算可能性的流程图

由某种图像采集设备(如摄像机)所取得的灰度(或彩色)图像经数字化后通过初始略图、 $2\frac{1}{2}$ 维描述、三维物体的表示(如用广义圆柱体)等三个层次的信息处理过程,最后得到视觉理解的结果。初始略图的计算是对灰度图像中的对象提取其基本轮廓(或者说是图像中被研究对象的边缘),使被研究的对象与复杂的图像背景分离开来。这里,D. Marr 提出采用高斯拉普拉斯滤波再求过零点的方法,它具有多尺度滤波的特点。求 $2\frac{1}{2}$ 维描述的过程把隐含在二维图像中有关对象的三维信息提取出来。例如,在光照等条件相同的情况下,可以认为图像中物体表面的明暗度变化是由表面的不同取向所导致的,因而这种明暗度变化中包含着有关表面形状的信息。通过计算得到的是对象各单元表面(占据一个像素大小的面积)的取向,这是介乎二、三维之间的一种信息(因为取向不是绝对深度),因而名之为 $2\frac{1}{2}$ 维描述。

通过前面两个计算步骤,可能得到图像中被研究对象的外轮廓和表面起伏,这就为下阶段的符号化提供了条件。通常采用的一种广义圆柱体(其截面形状和轴线走向可以改变的三维几何图形)的表示方法可以把三维物体分解成广义圆柱体的组合,从而实现对象的符号化并把它变成标记的集合。最后,通过对标记集合的匹配分析便可达到视觉理解的目的。

这一流程仅仅说明了进行视觉计算的可能途径,作为一个通用的计算机视觉系统,这种计算实现还存在着很大的困难。解题的病态性(可以建立的数学方程数与求解的数量不相适应和要求具有与解直接有关的先验知识等)与数据量很大的运算要求的错综结合使解的存在性和惟一性发生了问题,并形成极为专门的应用数学课题。在继续进行理论探索的同时,为取得实用的效果,人们探讨了多种有局限性的应用途径,例如引入物理附加约束条件的主动视觉理论(参见计算机视觉)就是一个很有实用意义的方向。

参考文献

1. Marr D. 著. 视觉计算理论. 姚国正等译. 北京: 科学出版社, 1987
2. 李介谷. 计算机视觉的理论和实践. 上海: 交通大学出版社, 1991 (李介谷)

shipin guangpanji

视频光盘机 (video compact disc player)

将视频光盘上存储的信号复现的设备,是只读光盘驱动器的一种。简称视盘机。

视频光盘有三大类:激光视盘(LVD)、CD视盘和DVD视盘。

LVD盘 已在家庭和商业中应用多年,主要用来录制电影或电视节目。盘外径有300 mm和200 mm两种。用压印的工艺制成“凹坑”和“平台”来存储信息(参见只读光盘驱动器),形成的光轨距离是1.67 μm 。对于采用NTSC制的电视节目,单面(300 mm)共有54 000条光轨,当采用等角速度(CAV)时,盘转速为1 800 r/min,这时每条光轨记录的容量相当于电视上一帧的内容,一张LVD盘可放映30分钟。当采用等线速度(CLV)工作模式时,线速度为10~11.4 m/s,单面可放映60分钟。LVD盘采用频率调制方式编码。

CD视盘 有几种规格:CD-V、CD-I、CD-G、VCD和SVCD。

CD-V盘 有三种规格:CDV-LP盘的直径是12英寸,可存储2小时的影像资料;CDV-EP盘的直径是8英寸,可存储40分钟影像资料;CDV-single盘的直径是5英寸,可存储20分钟音响加上5分钟图像资料。CD-V盘与LVD盘不同的是,它的音乐、文字等信息是用只读光盘(CD-ROM)的格式存储的,即采用的是8/14调制编码方式。

CD-I盘 是所谓“交互式激光光盘”。它采用CD-ROM的格式,声音、图像、文字和数字可混合存储。可与计算机连接,也可直接输出到电视机,因此应用广泛。在CD-I光盘机上可使用CD-ROM。

CD-G盘 的直径为120 mm。它的声音和数据的记录格式同CD唱盘,不同的是在CD唱盘的R-W副码区记录图形控制信息,因此在屏幕上看到图形和文字的同时能听到音乐和声音。一张CD-G盘可存储2 000幅16色彩色图像。

VCD盘 采用MPEG1压缩技术,图像清晰度在250线左右,两声道伴音,在120 mm盘上可存储74分钟的影视节目。VCD视盘机可播放VCD视盘、CD唱盘,但不能播放LV激光视盘及CD-V、CD-G视盘。

SVCD盘 采用MPEG2压缩技术,图像水平清晰度为350~400线,拥有4声道或双立体声,在120 mm盘上可存储35分钟到74分钟的影视节目。SVCD视盘机可播放SVCD视盘、CD唱盘和VCD视盘。

DVD盘 (参见数字多用途光盘) 可以分为单面、双面结构,每面可有一层或多层记录。普通的

双面光盘结构是将两张直径为 120 mm、厚度为 0.6 mm 的盘面背对背粘合成,其最小的凹坑长度为 0.4 μm ,道间距 0.74 μm ,采用 635 ~ 650 nm 的红外激光器读取数据。采用 MPEG2 压缩技术,图像水平清晰度达 500 ~ 1 000 线,提供 6 个完全独立的声音,单面单层 DVD 记录层具有 4.7 GB 容量,若以接近于广播级电视图像质量需要的平均数据传输速率 4.69 Mb/s 播放,能够存放 133 分 20 秒的整部电影。双面双层光盘的容量高达 17 GB,可以容纳 4 部电影。DVD 采用 RS-PC 纠错编码方式、8/16 信号调制方式,以及有效的防软件被复制措施。DVD 视盘机可播放 DVD 视盘、CD 唱盘、VCD 视盘、SVCD 视盘。

视频光盘采用数字化记录方式,图像画面质量高,音响保真度好,存储量大,适合存储电影、音乐、游戏以及各种有图像又有文字的资料。光盘表面每个光轨都有地址,具有快速搜索、显示帧号、播出静止画面等功能。视频光盘机虽然规格很多,但在结构上与只读光盘机类似,分成读出光盘头、寻道控制、主轴稳速控制等几个部分。

视频光盘机已成为广泛使用的家庭音像设备,同时也可作为多媒体计算机中的一种外围设备。另外,还有各种小型化的视盘机、自动换盘的多盘片视盘机以及多功能视频播放机,如可以上网、玩游戏的 VCD、SVCD、DVD 视频播放机等。

参考文献

1. Isailovic J. Video disc Systems: Theory and Applications. New Jersey: Prentice-Hall Inc., 1987

2. Arnd B. Real Time Recording of Video Compact Disks. Computer Engineering Thesis, Espoo-Vantaa Institute of Technology, Finland, Aug. 2001

3. Scott J. Video in the 21st Century. California: Intel Press, Mar. 2002

(余胜生)

shipin huiyi

视频会议 (videoconferencing) 集语音、文字和图像通信于一体的,用于多点实时交互式通信的一种多媒体通信技术。它可以将不同地点与会人员的活动情况、会议内容及各种文件资料以可视的形式展现在各个分会场。这是一种快速高效、应用日益增长的新的通信业务。

根据通信结点的数量,视频会议系统可分为:点对点视频会议系统和多点视频会议系统。

点对点视频会议系统支持两个通信结点间的视频会议通信功能,例如可视电话系统(参见可视电话)、分散在两个场地中的桌面视频会议系统或会议室型视频会议系统等。后者在会议室型视频会议系统的支持下,一群与会者集中在一间特殊装备的会议室中,与远地另一个会议室的与会人员进行交互通信,实现两点间的视频会议功能。由于会议室与会者较多,因此对视听效果要求较高,一套典型的系统一般应包括一台或几台大屏幕监视器、高质量摄像机、音响设备和控制设备等。

多点视频会议系统允许 3 个或 3 个以上不同场地的参加者同时参与会议。多点视频会议系统的一

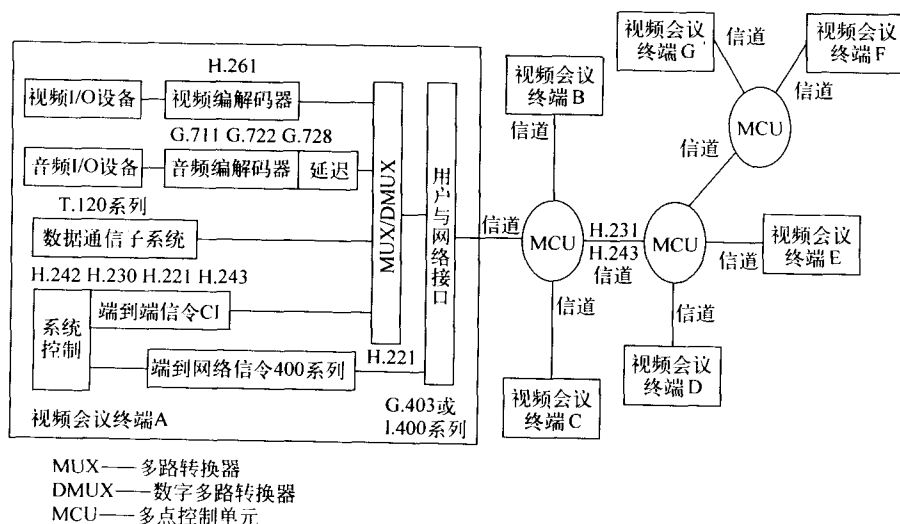


图1 视频会议系统结构框图

个关键技术是多点控制问题,多点控制单元(MCU)在通信网络上控制各个点的视频、音频、通用数据和控制信号的流向,使与会者可以接收到相应的视频、音频等信息,维持会议正常进行。图1是多点视频会议系统的结构框图。

视频会议的标准主要有国际电信联盟(ITU-T)制定 H. 320, H. 323, H. 324 及 T. 120 和个人会议工作组(PCWG)制定的个人会议标准(PCS)等。

(钟玉琢)

shipin sui ji cun qu cun chu qi xin pian

视频随机存取存储器芯片(video random access memory chip) 用于存储、传送和显示视频图像、文字信息的一种专用随机存取存储器芯片。简称 VRAM 芯片。VRAM 芯片内不仅有存储信息的随机存取存储器芯片,而且有适合于图形数据存储和显示的逻辑控制电路。

VRAM 芯片用其内部的动态随机存取存储器

(DRAM)芯片来存储一帧图像(由像素组成的阵列)中每一像素的颜色码或其颜色在颜色表存储器中的地址,以及其他与图像、文字处理有关的信息。根据颜色地址,由颜色表存储器和数模转换器将像素的红、绿、蓝三个基本色的强度转换为相应的电平值。

VRAM 芯片通常有帧存储器芯片和多端口视频随机存取存储器芯片两种。

帧存储器芯片由存储一帧图像的像素颜色表地址码的 DRAM 和存储一行像素颜色表地址的行缓冲寄存器组成。行缓冲寄存器由可高速读写的静态随机存取存储器构成。一行像素的颜色表地址串行写入输入行缓冲寄存器,再写入 DRAM;或从 DRAM 读出一行像素的颜色表地址,先送到输出行缓冲寄存器,再串行输出。帧存储器芯片主要用于电视和录像机。不同厂家、不同型号的帧存储器其结构也各有差异,主要体现为 DRAM 的构成和地址、命令输入方式等不同。

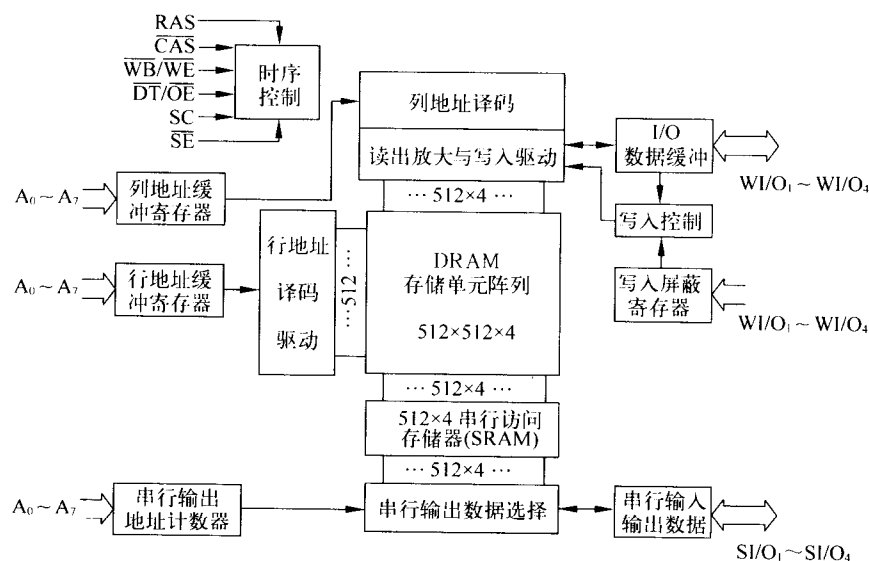


图1 256kb x 4 双端口 VRAM 芯片框图

多端口视频随机存取存储器芯片多用于计算机视频图像显示系统中,一般使用的是双端口 DRAM。双端口分别是可随机访问的 DRAM 端口和可串行访问的串行存取存储器(SAM)端口。而帧存储器芯片通常只有串行访问(输入输出)端口。

DRAM 端口除了可进行通常的 DRAM 操作外,增加了按位写入的功能。即 DRAM 的 4 位或 8 位数

据输入时,可只写入其中一位或某几位,其他位不写入,而保持原有的数据。这样就便于输入直线或曲线。新写入的线覆盖原有的图像,而不在线上的图像保持原状。DRAM 端口可通过图像控制电路与中央处理器(CPU)总线相接,便于 CPU 直接访问 DRAM。

SAM 端口是高速串行输入输出端口。SAM 端

口在外部时钟控制下实现像素数据的串行输入或输出。SAM 与 DRAM 间按行进行数据转换。串行输出主要用于屏幕显示的刷新,串行输入则是通过控制电路高速输入一行数据,再转换写入 DRAM。

由此可见,用 VRAM 芯片比用普通 DRAM 芯片作显示存储器可更有效地改写屏幕,提高了图像变换的速度,并且也允许有更高的像素频率(更高的分辨率)。但 VRAM 芯片的成本高于普通 DRAM 芯片。

双端口 VRAM 芯片有 $64\text{ kb} \times 4$, $256\text{ kb} \times 4$, $128\text{ kb} \times 8$, $256\text{ kb} \times 16$ 等品种。图 1 为 $256\text{ kb} \times 4$ 双端口 VRAM 芯片框图。此 VRAM 芯片的核心是点阵为 $512 \times 512 \times 4$ 的 DRAM 和 512×4 的 SAM。其组成还有用于控制 DRAM 访问的行、列地址缓冲寄存器、行地址译码驱动、列地址译码、读出放大和写入驱动、输入输出(I/O)数据缓冲、写入屏蔽寄存器

和写入控制等外围电路。此外,还有用于控制访问 SAM 的串行输出地址(指针)计数器、串行输出数据选择和串行输入输出数据寄存器。

地址输入 $A_0 \sim A_7$ 确定访问 DRAM 的行地址、列地址,或访问 SAM 的起始地址。 $WI/O_1 \sim WI/O_4$ 为 DRAM 端口的数据输入输出, $SI/O_1 \sim SI/O_4$ 为 SAM 端口的数据输入输出, \overline{RAS} 和 \overline{CAS} 为行选和列选信号,SC 为串行数据输入输出控制时钟, \overline{SE} 为串口使能控制。 $\overline{WB}/\overline{WE}$ 为 DRAM 按位写入和数据写入控制。 $\overline{DT}/\overline{OE}$ 为 DRAM 与 SAM 间数据转换和读出时输出控制。

DRAM 端口的操作及 DRAM 与 SAM 间的转换根据 \overline{RAS} 处于下降边时, \overline{CAS} , $\overline{DT}/\overline{OE}$, $\overline{WB}/\overline{WE}$ 和 \overline{SE} 的状态确定,如表 1 所示。

表 1 双端口 VRAM 芯片的操作

RAS 处于下降边时				DRAM 端口的操作	SAM 端口的状态
\overline{CAS}	$\overline{DT}/\overline{OE}$	$\overline{WB}/\overline{WE}$	\overline{SE}		
L	X	X	X	列选在前的刷新操作	可按原传输方向读出或写入 SAM
H	H	H	X	正常读写操作	
H	H	L	X	按位写入操作	
H	L	H	X	读转换操作(DRAM→SAM)	SAM 转为读出
H	L	L	H	伪写入转换	SAM 转为写入
H	L	L	L	写入转换(SAM→DRAM)	SAM 可继续写入

注: H——高电平; L——低电平; X——无关系

随着半导体技术和计算机技术的发展,图形显示控制芯片的功能在不断增强,VRAM 芯片也在不断发展。除了存储容量(每芯片的字数和字长)扩大,读写速度加快(读出时间减小,工作频率提高)外,在上述基本功能的基础上,又增加了新的功能,开发了新的品种。增加的新功能有:①快擦除功能。同一行的各个单元写入同一个数据,以加快屏幕的擦除功能。②按块写功能。一行中列地址连续的 4 个地址的 DRAM 单元,根据列选控制和颜色寄存器的数据,同时写入这 4 个地址,以加快窗口清除和填充的操作。③SAM 的分割功能。SAM 分成两半可分别与 DRAM 转换数据,以便 SAM 的内容与屏幕像素间更好地匹配。

随着显示器分辨率的提高和颜色的丰富(真彩色),屏幕刷新要求更高的像素频率;更换图像时,显示存储器与 CPU 间需要更多的数据交换。为此,

双端口 VRAM 用 SAM 高频串行输出满足高像素频率的要求。并且一次 DRAM 访问,即可将一行数据送入 SAM,这样就减少了屏幕刷新占用 DRAM 的时间,允许 CPU 与 DRAM 间有更多的时间作数据交换。VRAM 增加的一些特殊工作方式,如按位写入功能、快擦除功能、按块写入功能等,又增加了 CPU 与 DRAM 间数据交换的效率。用 VRAM 比用普通 DRAM 作显示存储器有更高的图像变换速率,因而 VRAM 特别适用于高分辨率的图像处理显示控制卡。

参考文献

Prince B. Semiconductor Memories. 2nd ed. Chichester: John Wiley & Sons, 1991 (孙祖希)

shitu chazhi

视图插值(view interpolation) 通过对基准图

像中对应像素进行插值从而产生新的景物视图的图像生成方法。该方法着重于提高图形学中景物的绘制速度,它基于图像而不是传统的三维几何模型来表示景物,从而使得图像的绘制速度不依赖于景物的复杂度。视图插值方法要求新视点位于两基准图像视点所决定的直线(称基线)上,于是新视图可由基准图像线性插值产生,类似于图像变形技术。当然,如果有很多幅基准图,也可通过一系列的插值得到一定范围内任意视点的图像。

视图插值的方法假设已知基准图像中像素的深度值、相机的位置和朝向,因此可以自动计算出基准图像对之间的密集像素对应关系,再通过对对应点像素坐标的线性插值将基准图像“变形”到目标图像。只要视点的变化较小,这一插值机制一般可得到合理的结果。

视点和对象的运动往往造成可见性的改变,一种简单情形下的可见性解决方案是,假设所有基准视点都朝着相同的目标,并且生成图像的视点位置被限制在视角 90° 的变化范围内,在此假设下对象的可见性优先级可保持不变。

参考文献

1. Chen S E and Williams L. View Interpolation for Image Synthesis. In: Computer Graphics (SIGGRAPH'93), 1993, 279 ~ 288
2. Beier T and Neely S. Feature-based Image Metamorphosis. Computer Graphics, 1992, 26 (2): 35 ~ 42
3. 石教英主编. 虚拟现实及实用算法. 北京: 科学出版社, 2002 (徐升)

shiying xiezheng lilun

适应谐振理论 (adaptive resonance theory, ART) 从心理学、认知科学和神经生理学等角度出发,充分借鉴人脑工作时的特点,以竞争学习和自稳机制为原则所建立的神经网络模型。

Stephen Grossberg 一直从事用数学来描述人的心理和认知活动的研究,企图为人类的心理和认知活动建立一个统一的数学理论。1976 年, Grossberg 提出了适应谐振神经网络模型。它的思想是只有当新的输入向量与已存入记忆中的某个向量足够相似时,两者才能相互匹配,对有关的权值系数进行调整,从而使长期记忆得以改变,达到一种适应谐振状态。根据 Grossberg 提出的适应谐振理论神经网络模型,已经建立了三种人工神经网络模型,即 ART-

1, ART-2, ART-3。它们在功能及可实现性方面形成了一个不断完善的系列。ART-1 网络是由 Gail A. Carpenter 和 Grossberg 于 1986 年提出的一种两层网络,该网络是建立在 Grossberg 的 ART 原始概念之上的。它的输入观察向量是二值的。ART-1 网络可用作最邻近分类器,能存储任意数目的二值模式。ART-2 神经网络的输入可以是任意的模拟向量,其基本设计思想仍采用竞争学习的机制,系统由“特征表示场”及“类别表示场”两个层及两者之间的学习层组成。ART-3 神经网络是在 ART-2 的基础上发展起来的,它的突出点之一是便于在一个具有任意多个层次的复杂层次结构中嵌入按统一标准构成的 ART 模块。ART-3 神经网络的构成取决于两个关键问题。第一是找到一种对各层都适用的场结构,可以称之为“同形”结构。第二是找到一种两个场的神经元之间信息传递的算法。

适应谐振神经网络模型的基本结构如图 1 所示,由两个相继连接的存储单元 STM- F_1 和 STM- F_2 组成,分成注意子系统和定位子系统。 F_1 与 F_2 之间的连接通路为适应长期记忆。

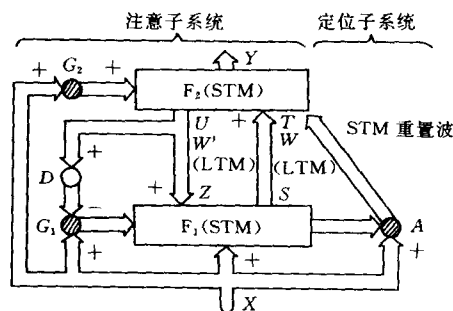


图 1 ART 结构示意图

适应谐振神经网络的工作过程可分为以下几个主要部分:

(1) 由下而上的自适应滤波和 STM 中的对比度增强过程。经预处理的信号 X 进入注意子系统的 F_1 的输入端,经 F_1 的节点变换成激活模式 S ,这一过程起到特征检出作用, F_1 中激活较大的节点就会有输出到 F_2 的信号,这就形成了输出模式 S 。 S 经过 F_1 到 F_2 之间通道时被 LTM 加权组合,变换为模式 T 而作用于 F_2 的输入。 S 到 T 的变换称为自适应滤波。 F_2 接受 T 后通过节点间相互作用迅速产生对比度增强了的模式 Y ,并存于 F_2 中。

(2) 自上而下的模板匹配和对已学编码的稳

定。一旦由下而上的变换 $X \rightarrow Y$ 完成后, Y 就会产生自上而下的激活信号模式 U 并向 F_1 输送。只有足够大的激活才会向反馈通道送出信号 U 。 U 经加权组合变换为模式 Z , Z 称为自上而下的模板或学习期望。现在 X 和 Z 两组模式作用于 F_1 , 它们共同产生激活模式 S^* 。一般说来 S^* 将会与只受 X 作用产生的 S 不同, 此时 F_1 的作用是要使 Z 与 X 匹配, 其匹配的结果决定了此后的作用过程。

(3) 注意子系统与定位子系统互相作用过程。输入模式在产生 S 的同时也激发了定位子系统 A , 但 F_1 中的 X 会在 A 产生输出前起禁止作用, 当 F_2 的反馈模式 Z 与 X 失配时, 就会大大减弱这一禁止作用, 当减弱到一定程度时, A 就被激活。 A 向 F_2 送出信号作用于它的全部单元将会改变 F_2 的状态, 取消原来的自上而下的模板 Z , 结束了 Z 与 X 的失配。于是输入 X 再起作用直到 F_2 产生新状态 Y^* , Y^* 产生新的自上而下模板 Z^* , 如果仍然与 X 失配, 定位子系统还会起作用。这样, 产生一个快速的一系列匹配与重置过程。此过程控制了 LTM 的搜索从而调整了 LTM 对外界环境的“编码”。搜索过程一直继续到 F_2 送回的模板与外界输入 X 相匹配为止。

(4) 注意增益控制及启动这一自上而下的模板匹配过程还有一些特点应予考虑。例如若在 F_1 送出由下而上的作用前 F_2 被激发, 此时 F_2 也会产生自上而下的模板 Z 作用于 F_1 , 这时 F_1 也会受到激发, 并产生自下而上的作用过程。注意, 增益控制会给出禁止作用来影响 F_1 对输入响应的灵敏度, 使 F_1 得以区分自下而上和自上而下的信号。

(5) 匹配。应用“2/3 规则”, 决定 F_1 的输出信号。当 X 结束时 F_1 的活动立即终止, 同时通过 G_2 的作用使 F_2 的活动也立即终止, 这就使 F_2 能够立即等待下一个输入向量的来到。

按照 ART 理论建立的神经网络是一种自组织神经网络, 采用多层双向结构。它在神经生理学及心理学等许多方面有模仿人脑神经系统工作的许多特点, 诸如层次性、双向性(由低向上和由顶向下), 注意力的集中和转移, 竞争选择和重置等等。这种系统毋需精确的定时, 对系统中的参数没有很高的精度要求, 具有很好的健壮性。在 ART 系统中采用了具有同场内神经元抑制性反馈的原理来选择最长的类别码, 形成了掩蔽理论。

目前, 适应谐振神经网络的研究主要是计算机模拟实验, 尚未见到硬件实现的系统。这种系统比

较复杂, 实现有些困难。它被用在图像识别、语音识别和语音生成等领域。

参考文献

史忠植. 神经计算. 北京: 电子工业出版社, 1993
(史忠植 叶世伟)

shoufaqi

收发器 (transceiver) 在以太网中, 网络适配器和传输媒体之间收发信号用的部件, 这里的传输媒体可以是双绞线, 也可以是同轴电缆或光缆。

收发器的功能为: ①接收来自网络适配器上的信号, 然后送往传输媒体发送出去; ②检测发送的信号是否发生碰撞; ③检测传输媒体上是否存在信号; ④接收来自传输媒体的信号并转送至网络适配器。

图 1 示出了以太网上收发器连接的两种情况。对于图(a)的 10 BASE 5 型网络而言, 收发器是直接作为传输媒体的粗同轴电缆(直径为 10 mm)相连接的, 同时还通过一根最长可达 50 m 的收发器电缆与网络适配器的连接单元接口(AUI)相连。对于图(b)的 10 BASE 2 型网络而言, 收发器作为一个器件安装在网络适配器上, 然后通过 BNC 连接器与传输媒体, 即直径为 5 mm 的细同轴电缆相连。

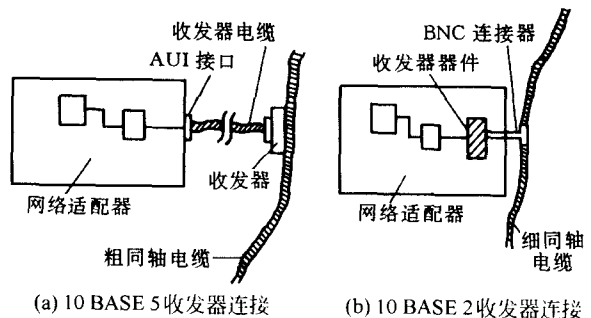


图 1 收发器连接图

(张公忠)

shoushi shibie

手势识别 (hand gestures recognition) 计算机识别人手(或手和臂的组合)姿态或动作并判断其意义的技术。其中人手(或者手和臂的组合)的各种姿态或动作称为手势, 包括静态手势(手的单个姿态、手形)和动态手势(动作, 由一系列姿态组成)。

基于视觉的手势识别其基本原理如图 1 所示。

系统通过输入设备获取图像序列(或视频流),检测其中是否有手势出现,如果有,则把该手势从图像序列(或视频流)中分割出来,然后进行手势分析,选

择手势模型并获取模型参数,最后对手势进行识别、分类形成关于手势的描述,从而得到最终的识别结果。

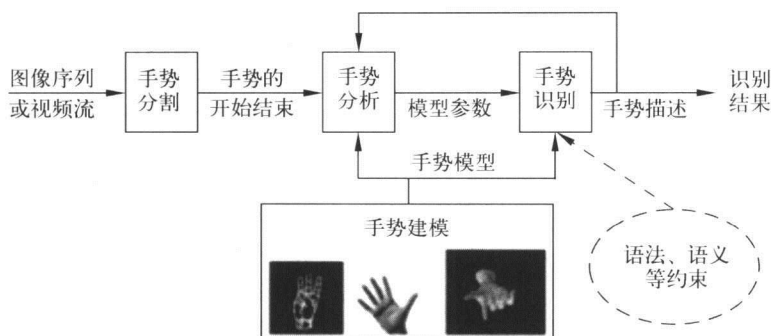


图1 基于视觉的手势识别基本原理框图(参考文献1)

手势模型的建立(手势建模),对手势进行分析确定模型参数(手势分析),以及最终的手势识别等是手势识别的3个重要组成部分,也是手势识别的技术关键。

手势模型对于手势识别至关重要,是指通过计算机提取出来的可以用来反映人手(或手和臂的组合)的静态表现(姿态)及动态运动特征(动作)的抽象化的模型。手势模型可以分成:①基于三维手

(臂)模型的手势模型;②基于表现的手势模型两大类,其中前者包括纹理模型、网格模型、几何模型以及骨架模型,后者包括基于灰度图像本身的表现模型、基于二维变形模板的模型、基于图像属性的模型、基于图像运动的表现模型等。图2举例给出了表示同一姿态的各种人手模型,可以在此人手模型的基础上建立相应的手势模型。

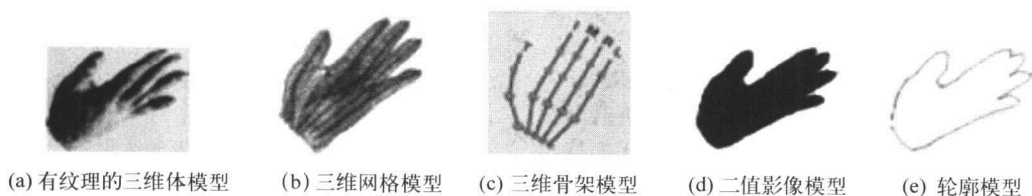


图2 表示同一姿态的各种人手模型(参考文献1)

手势建模是指通过分析手势在图像(序列)中的表现姿态或动态运动特征,并建立与其相对应的手势模型的过程,其目的是对表示各种不同意义的手势分类并建立模型。

手势分析的任务是估计选定的手势模型的参数,包括特征检测和参数估计两个部分。特征检测通过基于颜色定位、基于运动定位以及多模式混合的定位等技术手段检测图像(序列)中的人手位置;然后选择手势模型并估计模型的参数。

手势识别根据手势分析得到的模型参数(序列)识别其中的含义(意义)并得到关于手势描述的识别结果。根据静态手势识别或动态手势识别,其技术方法也有所不同。静态手势识别算法主要包括基

于经典参数聚类技术的识别、基于非线性聚类技术的识别等;动态手势识别由于涉及到时间和空间上下文的问题,因而比较复杂,常用的识别技术有基于隐含马尔可夫模型(HMM)、基于动态时间规整(DTW)以及基于压缩时间轴的识别等。

手势是一种自然直观的人际交流方式,利用计算机自动进行手势识别将大大增强现有的人机交互技术,实现更直接、更自然、更和谐的人机交互接口。手势识别在计算机远程操作、交互游戏、噪声环境中的交互交流等都会有广泛的应用。

手势识别的研究工作开始于1992年。近年来,手势识别技术获得了研究人员的普遍重视,国内外一些著名的研究机构和学校等都开展了这方面的工

作,目前已经取得了一些研究成果。然而,手势识别也还存在一些问题,比如光照、背景的影响等。另外,成功的手势识别策略应该考虑手势的时间和空间上下文,即考虑手势的语法规则。语法规则既要反映手势的语言学特征,又要反映手势的空间特征。手势识别还处于研究阶段,离真正进入实际应用还有较长的路要走。

参考文献

1. 任海兵, 祝远新, 徐光祐等. 基于视觉手势识别的研究综述. 电子学报, 2002, 28(2): 118 ~ 121
2. Hamdam R, Heitz F, and Thoraval L. Gesture localization and Recognition Using Probabilistic Visual Learning. In: Proceedings of the IEEE Comp. Soc. Conference on Computer Vision & Pattern Recognition, vol. 2, pp. 98 ~ 103, 1999 (吴志勇 蔡莲红)

shuchu shebei

输出设备(output device) 将计算机处理过的信息转变成其他机器能识别的或表现为人能理解的形式的设备,例如人能够看懂的显示在屏幕上或印刷在纸上的文字、符号、图形、图像或能够听懂语音的各种设备。常见的输出设备有显示器、印刷设备以及绘图机和言语(语音)输出设备。显示器的输出结果不能长期保存,而印刷设备和绘图机可用来长期保存文字、图形和图像的输出结果。

显示器常见的有 CRT 显示器、液晶显示器。

印刷设备是一种提供硬拷贝的计算机输出设备,它有许多种类型。根据印字头对纸有无击打动作可分为击打式打印机与非击打式印刷机两大类;按字形的表现方法可分为完整笔画的整字型印刷与用紧密邻接的点构成字形的点矩阵(点字符)型印刷两类;按印刷过程中逐字处理与逐行处理的方式又可分成串行印刷与并行(行式)印刷,以页为单位进行印刷的称为页式印刷。按颜色分有单色印刷与彩色印刷。

早期的计算机印刷输出设备基本上是利用机电原理的击打式打印机。这类设备的机械结构复杂,打印速度受机械的限制难以提高,且噪声大,存在机械磨损问题。击打式的优点是可使用普通纸,并可同时复印数份。非击打式印刷设备利用物理的(光、电、磁、热)或化学的方法实现印刷,而不是靠机械冲击的方法,因而噪声低,速度快。常见的有激光印刷机、喷墨印刷机、静电印刷机、热升华印刷机等。其缺点是不能同时印刷多份,有些非击打式印

刷设备还需用特殊的印刷纸。非击打式印刷机几乎都是点字符型,字型、字体变换灵活,适于图形、图像及汉字的印刷。

言语(语音)输出设备利用言语合成技术(参见汉语言语(语音)合成)来实现言语输出。言语输出应用于将文本中的文字转换为口语,以及交通(如公共汽车自动报站)、通信(如查号台自动应答)等方面。

将计算机的电子信息转变成其他机器所能识别(感知)的形式的设备亦是一类输出设备。例如联机的卡片穿孔机、纸带穿孔机、磁带机、软盘驱动器、光盘驱动器等均可视为这类输出设备。例如,通过联机磁带机或软磁盘驱动器等存储设备暂时将待输出的信息记录在磁带或软磁盘等媒体上,然后将这些媒体转移到由磁带机或软磁盘驱动器控制的脱机打印设备或绘图机上,进行脱机印刷或绘图。又例如利用联机卡片或纸带穿孔设备以孔的形式将输出结果保存下来,然后,用这些穿孔媒体去控制数控机床的运转。

计算机输出的数字量经数模转换器转换为模拟信号,再送往自动控制系统进行过程控制。数模转换器等亦可视为一类输出设备。(刘锡刚)

shuru shebei

输入设备(input device) 将待输入的数字信息或各种形式的信息转换成适宜于计算机处理的数字信息并送入计算机的设备。输入设备可分为两类:采用媒体输入的设备 and 交互输入设备。

采用媒体输入的设备有纸带输入机、卡片输入机、光学字符阅读机(OCR)、光学标记阅读机(OMR)、磁带驱动器、软磁盘驱动器、光盘驱动器等。这些设备把记录在各种媒体(如穿孔纸带、穿孔卡片、磁带、软磁盘、光盘、纸张、信息卡)上的信息送入计算机。一般为成批输入,输入过程中使用者不作干预。

交互输入设备指不使用记录信息的媒体,而由使用者通过操作直接和计算机进行信息交换的设备。常见的有键盘、跟踪球、鼠标器、触屏、数字化仪、光笔等。交互输入设备通常和显示器连用。使用者在输入时,从显示器的屏幕上可以看到输入的内容和计算机对输入所作出的反应,即输入是采用使用者和计算机相互对话的方式进行的。如用键盘编辑、修改程序,用控制杆玩电子游戏,用鼠标器操作、控制计算机等都是交互输入。在输入过程中,使

用者可以即时进行处理,如保留、删去、插入、修改、移动等。交互输入一般不强调快速性,而重视人机界面的友善性,即使用方便,对话自然等。

20世纪90年代以来,许多自然语言输入设备,如言语(语音)输入设备、文字输入设备和图像输入设备等,都得到应用。

言语(语音)输入设备是将人类的言语转换为计算机能接受的数字信号并加以识别的设备。

言语输入设备由以下几个部件组成:①话筒等拾音器将言语转换为模拟电信号。②模数转换器将经过放大的模拟电信号转换成数字信号。③计算机和言语识别软件,将以数字表示的言语信息与计算机内部所存储的言语模型进行比较,从而找出最佳匹配作为识别结果。这部分是言语输入设备的核心。

言语输入设备有广阔的应用领域,如言语听写机(言语输入代替键盘输入)、声控系统(使用声音进行自动控制)、电话的语音拨号(以说人名或单位名代替拨号)等(参见汉语言语(语音)识别)。

文字输入设备在我国主要是汉字识别系统。20世纪70年代末,我国开始对汉字识别进行研究。

20世纪80年代末是印刷体汉字识别和联机手写汉字识别的实用的开始。90年代以来,利用汉字识别技术研制成功的识别系统正逐步成为商品。

汉字识别包括以下类型:①**联机手写汉字识别**。用笔在图形输入板上写字,人一面写,机器一面认,是一种交互型汉字输入手段。设备由图形输入板(即小型数字化仪)和微型计算机组成。②**印刷体汉字识别**。识别已印刷在纸上的各种印刷体汉字,包括照排机、打印机输出和印刷的汉字。通常能识别宋、仿宋、楷、黑等各种印刷体,同时能识别西文和数字以及一些符号。印刷体汉字识别通常是脱机识别。设备由扫描仪和微型计算机组成。

随着多媒体技术的发展,除**图像获取**之外,**数字音频获取**与**数字视频获取**都成了计算机的重要技术,因此,相应的设备,如**扫描仪**、**数字相机**、**数字摄像头**和**数字摄像机**等也都成了计算机的常用输入设备。

交互输入设备是研究的热门之一。因为人机交互能力是影响计算机使用的一个重要因素。为了实现自然语言输入,使人与计算机之间能像人与人之间一样采用自然语言交谈,需要有更完善、更理想的交互输入设备。

(林兼)

shuru shuchu guanli chengxu

输入输出管理程序(input/output manager)

操作系统中用于组织和管理输入输出(I/O)设备,以完成输入输出操作的程序。它的主要任务是有效地处理对I/O设备的使用请求、实现I/O程序设计和I/O设备驱动调度,实现主存和外围设备的数据传输操作。通常,I/O设备、I/O控制部件和I/O管理程序统称为I/O系统。

设备独立性是对设备管理的基本要求。实现设备独立性的关键是把用户所用的设备与计算机中实际进行I/O操作的物理设备分离开来,系统通过逻辑设备来接受用户的I/O请求,而由设备管理程序来实现由逻辑设备到物理设备的转换。

在I/O系统中,普遍采用I/O中断、缓冲区管理、通道等多种技术,较好地克服了由于I/O设备和处理器在速度上的不匹配所引起的缺点,使主机和外设并行工作,显著改善了I/O设备的使用效率和系统性能。

对于磁鼓、磁盘等作为后援存储器的大容量辅存设备,同时会有许多I/O请求到达,具有繁重的工作负载,为了降低若干个I/O请求执行的总时间,从而提高系统效率,应该按最佳次序执行这些I/O请求,这称作**设备驱动调度**,使用的算法叫**驱动调度算法**。有效的驱动调度算法有:循环排序、优化分布、多重副本等。适用于磁盘的寻查定序算法有:先来先服务、最短查找时间优先、电梯调度算法、扫描算法等。

为了实现设备的分配和去配,系统应建立和维护一张设备状态表,表中每个表项应包括:设备类型、设备地址、占用进程、当前设备状态、请求使用设备的进程队列指针等。按设备使用方式可分为:独占设备、共享设备和虚拟设备。让一个作业整个运行过程中占用的I/O设备称**独占设备**,独占设备采用静态分配。共享设备允许多个用户同时使用,像大容量辅助存储器,一般不必进行分配,用户可通过文件系统按名存取共享设备中的信息。打印机可作为共享设备,这时采用动态分配,作业通过申请和(或)释放系统调用来获得I/O设备,系统则动态分给各作业使用。

虚拟设备所用技术称为**假脱机操作技术**,是一类物理设备模拟另一类物理设备的技术,是使独占设备变成共享设备的技术。其基本实现思想是:利用多道程序设计技术,在主机执行计算任务的间隙,

将成批作业及其数据输入到磁盘称之为输入井的缓冲区中;此后,由作业调度程序选择作业多道执行。作业使用数据时,不必再启动独占设备,而只要从相应输入井读取数据。同样地,作业执行中不必直接启动输出设备输出数据,而将作业的输出数据暂存到磁盘称之为输出井的缓冲区中,待作业执行完毕,由系统组织数据成批输出。实现 Spooling 系统的主要数据结构有:作业表、预输入表和缓输出表;Spooling 系统的主要组成部分有:预输入程序、缓输出程序和输入井、输出井管理程序。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989
(费翔林)

shuru shuchu jishu

输入输出技术 (input/output technique)

实现计算机主机与外围设备进行信息交换的技术。外围设备包括辅助存储器和计算机与外部世界交换信息的输入输出设备。

输入输出设备根据其在计算机系统中的作用可分为: ①**输入设备**: 主要有纸带输入机、卡片输入机、软磁盘输入机、光学字符识别设备(参见**光学字符阅读机**)、图形数字化仪(参见**图形输入设备**)、键盘输入设备、磁墨水字符识别设备和语音输入设备等; ②**输出设备**: 主要有打印机、绘图机、纸带机、纸卡片穿孔机以及字符图形显示器等; ③**终端设备**: 主要有会话型终端、智能型终端以及远程成批处理终端等; ④**脱机设备**: 指没有计算机控制也可完成数据编制、媒体变换等工作的设备, 主要有卡片穿孔机、纸带穿孔机、磁带-绘图机等; ⑤**其他设备**: 泛指在主机处理数据前后对数据进行加工的设备, 包括**模数转换器**、**数模转换器**、外围处理机、开关量输入输出装置以及各种检测设备。另外, 还有**通信控制器**, 通过它可以实现计算机和通信线路的连接。

根据中央处理机输入输出操作参与的程度, 计算机系统中输入输出的控制方法有 3 类, 即: ①**程序控制**, 包括无条件(同步)传送、条件(查询或异步)传送和中断传送; ②**直接存储器存取(DMA)控制**; ③**通道控制(I/O 处理机控制)**。

总线是计算机系统中各功能模块间传送信息的公共通路。各种**总线标准**的特性及应用也与输入输出技术密切相关。
(孙德文)

shuru shuchu jiekou

输入输出接口 (input/output interface)

参见输入输出设备接口。

shuru shuchu shebei jiekou

输入输出设备接口 (input/output device interface)

计算机主机与外围设备之间以及两个外围设备之间的交接部分。在计算机系统中, 计算机与外围设备之间应有特定的硬件连接和相应的软件控制, 以适应和协调不同外围设备的需要, 便于完成计算机与外围设备之间相互交换信息。这个硬件和软件的综合体称为输入输出设备接口, 简称 I/O 接口, 也称为**外围设备接口**。

早期的计算机系统主要是通过设计不同的控制电路来实现各种外围设备与主机的连接, 即以硬件接口设计为主。随着大规模集成电路的发展, 接口有专用芯片可供选用, 使接口技术进入标准化阶段。另一方面, 由于接口芯片向智能化方向发展, 接口中软件的比重也越来越大。许多情况下, 采用微处理器作外围设备的接口芯片, 如键盘的接口芯片就是一个**单片计算机**。

I/O 接口的功能、种类和工作方式

为了满足不同外围设备在不同速度、不同方式下的工作需要, 接口应具有以下基本功能:

(1) **地址译码和设备选择** 在系统中常有多个外围设备, 所以系统有多个与之对应的接口。例如, 在**微型计算机系统**中, **中央处理器(CPU)**用地址码选择外围设备接口。当一个接口里有多个端口时, CPU 对接口操作, 寻址的是端口, 因此接口本身就要有地址译码的能力, 以便使 CPU 对特定的端口操作。

(2) **数据缓冲和锁存** 数据总线是系统各部件信息交换的公共通路, 因此它不允许被某一部件长期占用, 特别是被相对速度比较慢的外围设备长期占用。因此数据不允许直接由数据总线送给**输出设备**, 也不允许直接由**输入设备**送到数据总线。这时接口就要发挥数据缓冲和锁存的作用, 将数据先锁存起来, 以后再发送, 这样就可解决主机与外围设备速度不匹配的矛盾。

(3) **信息变换** 当 CPU 与用串行方式传送数据的外围设备连接时, 就要求接口具有并串转换的能力。当输入输出信号是模拟量时, 要求接口具有**模数(A/D)**、**数模(D/A)**转换的能力。

(4) 传递控制指令和状态信息 主机和外围设备进行数据交换前,要确信设备是准备就绪的。因此,要对外部设备进行查询与控制,包括操作时序的协调、中断请求与响应、直接存储器存取(DMA)请求与响应、主机命令与输入输出设备状态的交换与传送等。

输入输出设备接口的种类很多。从主机的形式分,有小型计算机接口,微型计算机接口,大、中型计算机接口和分布式计算机接口。从应用角度分,有运行辅助接口,用户交互接口,传感接口和控制接口。从接口结构及功能选择的灵活性分,有可编程接口和不可编程接口。从接口的数据传送方式分,有并行接口和串行接口。从接口的通用性分,有通用接口和专用接口。按输入输出的信号分,为数字接口和模拟接口。图1示出了主机、输入输出设备接口以及外围设备之间的连接关系。

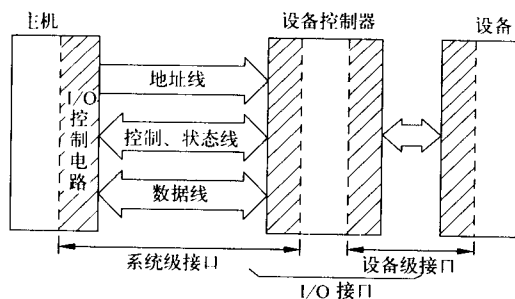


图1 主机、输入输出设备接口及外围设备的连接方法

由图1可见,输入输出设备接口应有两个连接方向的接口。一个是主机与设备控制器之间的接口,它控制外围设备与主机总线之间交换数据,称之为系统级接口,如常见的小计算机系统接口(SCSI)、智能外围接口(IPI)、外围设备互连(PCI)等。另一个是设备控制器与设备驱动器之间的接口,它根据主机的命令管理外围设备的操作,称之为设备级接口,如工业标准磁盘接口(ST 506/412)、增强型小设备接口(ESDI)、存储模块设备接口(SMD)、智能设备接口(IDE和EIDE)、通用串行总线接口(USB)、IEEE-1394接口和蓝牙接口等。

输入输出设备通过接口与主机进行信息传送的工作方式主要有四种:程序控制方式、程序中断方式、直接存储器存取方式和输入输出处理机(通道)方式。

大、中型计算机因主机速度较高,操作系统对主

机资源的管理能力较强,其输入输出设备接口主要采用多路独立通道结构。

分布式计算机通过网络连接的外围设备的数量一般较多,它主要是利用多路通道的分时控制功能来分时处理各外围设备交换的数据。因此,主机与外围设备的接口是通过多路通道连接实现的。

在小型计算机和微型计算机中,输入输出接口一般多采用程序控制、程序中断和DMA三种控制方式。这些方式中接口的硬件都很简单。它们在不同的软件支持下,可以和多种外围设备连接,前两种方式主要适用于低速设备,DMA方式主要适用于具有实时要求的外围设备和高速设备。

随着计算机技术的发展,输入输出设备接口技术已成为有效增强计算机效率与功能的重要环节,它的发展趋势是更加标准化、系列化和通用化,同时接口芯片向可编程发展,以扩大接口芯片的功能。

串行接口 计算机与外围设备之间按顺序逐位进行数据传送的一种通信接口。串行接口主要用于远程通信和低速输入输出设备。

串行接口通信方式有两种,一种是异步通信方式。在这种方式中,一般以一帧为一个单位传送,一帧由起始位、数据位、奇偶检验位、结束位等部分组成。用串行异步通信方式传送信息时,发送端和接收端之间必须约定数据格式,否则就会产生错误。另一种是同步通信方式,发送端与接收端之间用同步脉冲进行同步数据通信。

常见的串行接口有RS-232C、USB接口、IEEE-1394接口等。

RS-232C 常用的串行通信接口标准。它是1969年由美国电子工业协会(EIA)制定的。RS-232C用于一端是数据终端设备(DTE)(如计算机或数据终端),另一端是数据通信设备(DCE)(如调制解调器)的串行数据传送。RS-232C标准规定:不加调制解调装置时最长传输距离为100英尺(1英尺=0.3048m)。一般使用时不超过15m。信息传送速率最大为19200波特。

RS-232C采用25线的D型连接器。当不使用调制解调器直接连接时,只要5根线,使用很方便。

RS-232C采用负逻辑。逻辑0为+5~+15V,逻辑1为-5~-25V。

在微型计算机中常用的可编程的串行接口芯片有Intel 8250、Intel 8251、MC 6850等。

USB 接口 由 Compaq, HP, Intel 等公司提出的。1996 年制定了 USB 1.0 标准;2000 年制定了 USB 2.0 标准。USB 接口主要特点有:自动检测和配置的即插即用功能,带电热插拔操作,专用的标准接插件,支持菊花链式串行连接,一台计算机在一条通道电缆上可同时接入 127 个外围设备。

USB 接口的基本硬件有 USB 主控制器(或根集器)、USB 集线器、USB 设备等;通用软件有 USB 设备驱动程序、USB 驱动程序、USB 控制器驱动程序等。主控制器(或根集器)在主机方,下接集线器或设备。集线器有一个上行端口,最多可有 8 个下行端口。下行端口可接设备也可接集线器。这种层次结构使用非常方便。

USB 采用专用连接器和电缆。有 4 根线,其中两个用于传输差分信号,两个用于给 USB 设备供电。但 USB 设备也可以由单独的电源供电。

低速设备和高速设备所用的传输电缆在电气特性方面是不同的。对于低速外围设备,例如鼠标器和键盘,采用不带屏蔽的非双绞线。高速设备的电缆不仅要求屏蔽,而且两条差分数据线必须是双绞线。

USB 的传输类型有 4 种:

(1) 中断传输 由 USB 设备发出中断请求,传输的方向总是从 USB 设备到主机。

(2) 控制传输 用于配置 USB 设备,并对它操作的某些方面进行控制。

(3) 同步传输 要求恒定传输速率的实时应用。如用于音频传输(如 CD-ROM)。

(4) 批量传输 用于对数据传输速率没有特殊要求的设备,如打印机。

USB 使用差分信号在根集器和设备之间进行信息的串行传输。USB 2.0 标准规定接口数据传输速率:高速为 480 Mb/s,全速为 12 Mb/s,低速为 1.5 Mb/s。

USB 技术具有开放性,可用于台式机,也可用于便携机。可用于硬盘机、光盘机等存储设备,也可用于鼠标器、控制杆、键盘、显示器、打印机及其他输入输出设备(如数码相机)。在数字图像、电话语音合成、交互式多媒体、消费电子产品等应用领域也得到了广泛的应用。

IEEE-1394 接口 是 Apple 公司和 TI 公司开发的高速串行接口标准。原型称 Fire Wire 总线,1995 年由 IEEE 采纳和重新规范。

IEEE-1394 通过一条 6 针线缆将不同的设备连

接起来。在 6 针线缆中,两条双绞线用于数据传输,一条为地线,一条用于电力供应。数据是通过双绞线以分组的方式进行传送的,以减弱线缆中的噪声来实现高速数据传输,在数据组中包含有传送的数据信息和相应设备的地址信息。标准规定线缆的长度不超过 4.5 m。

IEEE-1394 标准定义了两种总线模式,即 Backplane 模式和 Cable 模式,其中 Backplane 模式支持 12.5 Mb/s, 25 Mb/s, 50 Mb/s 的传输速率;Cable 模式支持 100 Mb/s, 200 Mb/s, 400 Mb/s 的传输速率。IEEE-1394 接口具有把一个信息源传来的数据向多个设备输出的功能,特别适合于家庭视听 AV 设备的连接。

每一个支持 IEEE-1394 标准的设备都具有输入输出接口,用户可以采用方便的节点串联方式一次性连接最多 63 台不同的设备,IEEE-1394 不要求 PC 作为所有接入外设的控制器,不同的外设之间可以直接相互交换信息。另外,采用 IEEE-1394 接口,两台 PC 机还可以共用同一个外设,这是 USB 或其他接口都无法实现的。

IEEE-1394 接口支持热插拔。

并行接口 计算机与外围设备之间进行并行数据传送的一种接口,具有在多条线上一次同时传输一组二进制位的能力,通常能将给定字节(或者字)中数据的所有位使用各自的数据线同时传输。并行接口常用于快速、近距离的设备与主机的连接,并行接口传输效率高,但设备费用较高。

常见的并行接口有打印机 Centronics 接口、IEEE-488 接口、数模转换器和模数转换器接口等。

打印机 Centronics 接口 是打印机通用的并行接口,每次并行传送 8 位数据,最大传输距离 5 m,最大数据传输速率为 150 KB/s,采用单向传输。接口标准采用 36 针 D 型插头座。信号线中除了有数据线、控制线、状态线、地线和电源线外,还有一些空线,这些空线可供用户安排特别的用途。

EPP 接口 是 Centronics 接口的改进型。它采用双向、半双工传输,最大数据传输速率可达 2 MB/s,仍使用 36 针 D 型插头座。EPP 接口广泛使用于扫描仪。

IEEE-488 接口 也称为 HPIB 接口或 GPIB 接口。它最初是 HP 公司提出的,1975 年被接受为 IEEE 标准。

IEEE-488 标准采用 24 线的连接器,其中有 8 根双向数据线,3 根信号交换线,5 根控制线。以低

电平(小于+0.5 V)为逻辑1,高电平(大于+2.0 V)为逻辑0。总线上最多可连接15个设备,设备分成三类:控制方、发送方和接收方。数据传送时只有一个发送方,但可多个接收方。为使快慢不同的设备能一起工作,采用应答方式。当各方都准备就绪后,控制方通知发送方发送数据;当最慢的一个接收方接收完成后,发出应答信号,发送方发出信号通知控制方和所有的接收方,完成一次传送。

IEEE-488接口的最大数据传输速率为1 MB/s,最大信息传输距离为20 m。

IEEE-488有专用芯片,如Intel 8291, Intel 8292, Intel 8293等。

管理程序 控制硬件工作的软件。

一台计算机要正常工作,单有微处理器是不够的。主板上还有其他芯片,外围还有许多设备,内存与外围部件的数据传输,与外界的联系通道等,都需要管理。在个人计算机中这些硬件的管理者称为基本输入输出系统(BIOS)。

BIOS中包含有输入输出设备的驱动程序,这部分程序实现对显示器、磁盘、键盘、打印机和通信口的驱动和控制。各个程序模块执行不同的功能,可通过中断调用。BIOS对系统中的主要输入输出设备提供设备级的控制,因此是PC系统中最靠近硬件、最底层的软件。

除了硬件的管理工作之外,BIOS还负责开机的例行检查、开机时进行初始化和各项功能设置、操作系统引导等工作。

BIOS存放在只读存储器中,它不致因电源关闭而消失。这种固定存放有序指令和数据集合的电路称为**固件**。换言之,固件是软件与硬件的结合体。固件除了采用只读存储器外,也可采用可擦编程只读存储器和可编程只读存储器(参见可擦编程只读存储器芯片和可编程只读存储器芯片),通过改写来升级。

参考文献

1. 聂丽文等. 微型计算机接口技术. 北京:电子工业出版社,2002
2. 袁新燕等. 计算机外设与接口简明教程. 北京:北京航空航天大学出版社,2000 (林兼)

shuru shuchu tongdao

输入输出通道(input/output channel) 主存储器与外围设备之间信息传输所使用的物理通道。它能同时管理多台外围设备的工作和信息传

输,以减少主机直接对外围设备的管理,实现数据处理的并行工作。

只作数据输入的通道称为输入通道,只作数据输出的通道称为输出通道。

由于外围设备种类繁多,传送速度和传送方式各不相同,为了提高控制效率,可采用不同的通道结构,如选择通道、成组多路通道、多路转换器通道和字节多路通道等。

选择通道 每次只选择1台高速外围设备(磁带、磁盘等),以成组方式与主机交换数据。选择通道虽可连接多台设备,但每次只能选择其中的1台。

成组多路通道 可控制多台高速外围设备以成组交叉方式分时传送数据,某台设备与主机交换数据时,允许别的设备作寻址或其他准备传输的操作,实现寻址和传送数据并行工作,以充分发挥通道高速交换数据的能力。成组多路通道在一次传输期间能传输大量的数据,是大型计算机、巨型计算机经常采用的数据传输方式。

多路转换器通道 可使计算机与若干低速外围设备(终端和打印机等)同时传输数据。多路转换器通道采用数据交叉的方法同时传输多台外围设备的数据,有时也称**交换器**。

字节多路通道 连接多台低速外围设备(行式打印机、读卡机、远程终端等)以字节方式交叉传送数据,多台低速设备分时占用通道和主机交换数据。一个字节多路通道可连接多个子通道,每个子通道可连接1台或多台设备,不同子通道的设备可同时工作,相同子通道内1次只允许1台设备与主机交换数据。

(徐良贤)

shubiaoqi

鼠标器(mouse) 一种控制显示器屏幕上光标位置的输入设备。在驱动程序的支持下,鼠标器能完成图形用户接口的功能。第一个用于IBM-PC的鼠标器是1982年由Mouse System公司研制成功的。由于它操作方便,易于使用,随着UNIX, Windows等带有图形用户接口的操作系统的普及,鼠标器已成为个人计算机和 workstation 必备的输入设备。

鼠标器因其外形(如图1)而得名,它通过电缆与主机相连接。20世纪90年代以来,采用红外线、电磁波与主机通信的也有应用。

当鼠标器与计算机连通时,在显示屏上就会出现一个光标,通常是一个指向左上方的小箭头。程序所操作的区域就是箭头顶部所指的区域。当将鼠

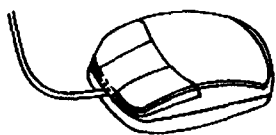


图 1 鼠标器

鼠标器在一个平面上移动时,其底部的传感器就把移动的方向和距离检测出来,送入计算机,控制光标作相应的运动。鼠标器上带有 2 个或 3 个按钮,有的还有上网专用按钮。移动光标对准显示在屏幕上的命令或图标,并按下按钮即可完成操作。

鼠标器是一种相对定位设备,不受平面上移动范围的限制。它的具体位置也和屏幕上光标绝对位置没有对应关系。当鼠标器脱离平面时,屏幕上光标并不会移动,直到鼠标器在平面的另一位置放下并再度移动时,光标才会跟着运动。

鼠标器的主要性能指标是动态分辨率和跟踪速度(或称辨识速度)。动态分辨率指鼠标器每移动一个单位距离能检测出的脉冲数,通常用点数每英寸 dpi 表示。鼠标器的动态分辨率有 200, 400, 800 dpi 等。跟踪速度指鼠标器不发生错误的最大允许移动速度,2003 年已达到 10 m/s。

鼠标器依照所采用的传感技术可分为机械式、光电式、机电式三种。

机械式鼠标器底部有一圆球,当鼠标器在平面上运动时,圆球与平面摩擦带动内部两个圆盘运动,圆盘上有编码器,通过接口把运动的方向和距离信号送入计算机,经软件处理,控制屏幕上光标作相应移动。机械式鼠标器结构简单,不需要专用平板,一般桌面或工作台面都可使用。但可靠性差,现在已很少使用。

光电式鼠标器采用光电传感器,底部不设圆球,而是一个光电元件和光源组成的部件。当它在专用的有明暗相间的小方格的平板上运动时,光电传感器接收到反射的信号,测出移动的方向和距离。光电式鼠标器分辨率高,可靠性高,但需要专用的平板。

机电式则是上述两种结构的综合。它底部有圆球,但圆球带动的不是机械编码盘而是光学编码盘,从而避免了机械磨损问题,也不需要专用的平板。

鼠标器与主机间的连接是通过把鼠标器接在通信口上实现的。个人计算机是把鼠标器接在串行通

信口 COM1 或 COM2 上,也可采用 PS/2 接口或 USB 接口。
(林兼)

shuxing wenfa

属性文法 (attribute grammar) 允许为每个终结符和非终结符配备一些属性的文法。它既能描述程序设计语言的语法,又为其语义处理提供了方便。属性文法由 D. E. Knuth 于 1968 年引进。属性有不同的类型,可以像变量一样地被赋值。赋值规则附加于语法规则之上,赋值与语法分析同时进行,赋值过程就是语义处理过程。在推导语法树的时候,诸属性的值被计算并通过赋值规则层层传递。有的从语法规则左边向右边传,有的从右边向左边传。语法推导树最后完成时,就得到出发符号的属性值,也就是整个程序的语义。

属性有两种,一种叫继承属性,另一种叫综合属性。继承属性的计算规则为由顶向下,对语法规则来说是从左至右。综合属性的计算规则为由底向上,对语法规则来说是从右向左。例如,在

$$A_p \rightarrow B_{q,r} C_{s,t} \quad (\text{语法规则})$$

$$[p = q + s, r = p + t] \quad (\text{属性赋值规则})$$

中, p, q, r, s, t 是属性,分别属于 A, B, C 三个非终结符,其中 p 是综合属性, r 是继承属性。 q, s, t 的性质要把其他语法规则中的属性赋值规则一起考虑才能确定。

规定: 每个语法符号的继承属性集和综合属性集之交为空。但即使在此前提下,属性及其赋值规则仍是不能任意给定的,否则会引起问题,如属性互相依赖,因而无法计算。D. E. Knuth 称没有循环依赖关系的属性文法为良义属性文法。然而加扎耶里等又证明了: 确定一个属性文法是否良义的任何算法必然具有指数复杂性。因此许多人致力于寻找良义属性文法的子集。如 U. Kastens 的有序属性文法和肯尼迪等的绝对无循环属性文法。

S. Fang 在 1972 年研制的 FOLD 系统中,首先实现了属性文法,其中使用了非确定性的属性计算方法。后来出现的用确定性方法计算属性文法的系统一般都有某些限制,如对扫描的遍数和扫描的方向加以限制。此外,著名的 L 属性文法适用于由顶向下的分析, S 属性文法适用于由底向上的分析。属性文法在编译程序的机械生成中起了很大的作用,使人们不仅可以机械地生成语法分析程序,还可以机械地生成语义处理程序。
(陆汝钤)

shu

树 (tree) 无回路的连通无向图。又称无向树 (参见图论)。每个分支都是树的无向图称为森林。如果树 T 是无向图 G 的生成子图, 则称 T 为 G 的生成树。如果森林 F 是无向图 G 的所有分支的生成树之并, 则称 F 为 G 的生成森林。如果森林 F 有 n 个顶点, m 条边和 k 个分支, 则 $m = n - k$ 。设 $\langle G, w \rangle$ 是加权图, $G' \subseteq G$, G' 中所有边的加权长度之和称为 G' 的加权长度。设 $\langle G, w \rangle$ 是加权连通无向图, G 的所有生成树中加权长度最小者, 称为 $\langle G, w \rangle$ 的最小生成树。

设 G 为弱连通有向图, 其中一个顶点的入度为 0, 其余顶点的入度均为 1, 则称 G 为有向树。入度为 0 的顶点称为根, 出度为 0 的顶点称为叶, 出度大于 0 的顶点称为分支顶点, 从根至任意顶点所经过的边数称为该顶点的级, 所有顶点的级的最大值称为有向树的高度。每个弱分支都是有向树的有向图称为有向森林。设 $m \in \mathbf{N}$ (自然数集合), D 为有向树, a 和 b 是 D 的两个顶点, 如果有一条边从 a 到 b , 则称 a 是 b 的父亲, b 是 a 的儿子。如果 D 的所有顶点出度的最大值为 m , 则称 D 为 m 元有向树。如果对于 m 元有向树 D 的每个顶点 v , 均有 $d_0^+(v) = m$ 或 $d_0^+(v) = 0$, 则称 D 为完全 m 元有向树。完全二元有向树又称为二叉树。在计算机科学中二叉树特别有用, 它可以用来研究算法的效率等问题。对于 n 阶二叉树 D , D 有 $(n+1)/2$ 个叶, D 的高度 h 满足: $\log_2(n+1) - 1 \leq h \leq (n-1)/2$ 。为每一级上的顶点规定了次序的有向树称为有序树。如果有向森林 F 的每个弱分支都是有序树, 且也为 F 的每个弱分支规定了次序, 则称 F 为有序森林。为每个分支顶点的儿子规定了位置的有序树称为定位有序树。

(张强)

shuju

数据 (data) 对象的表示, 即事实、概念或指令按照适合于通信、解释或处理 (借助人或自动装置) 的方式所形成的表示。

数据是计算机科学与技术中最为基本的概念, 几乎处处可见 (参见程序)。

以教师向学生传授知识为例, 教师可以通过形、声、文等方式向学生讲授某一概念, 相应的形、声、文只是表达概念的不同表示, 它们都是数据, 本身不具有任何含义。

参考文献

IEEE standard glossary of software engineering terminology. New York, N. Y. USA. IEEE, 1990

(徐家福)

shuju anquanxing

数据安全性 (data security) 数据库中为保护数据而具备的防御能力。它用以防止对数据未经授权的泄露、修改或对数据的有意与无意的破坏。

数据安全性是数据的拥有者及使用者都十分关心的问题。它涉及到法律、道德及计算机系统等诸多因素。这些因素可以分成两大类, 一类是与数据库系统本身无直接关系的外部条件, 另一类则是数据库系统本身的防范能力。

就外部条件而言, 它包括将数据按密级分类, 控制接触数据的人员, 对数据进行检验等一系列恰当的管理方针与保密措施。也包括对计算机物理环境, 设备的安全保卫与防辐射等手段。

就数据库系统本身的防范能力而言, 它包括数据库系统本身为数据安全所提供的各种措施。这些措施有: ①系统为管理人员提供授权手段以控制对数据库的访问。②允许对用户进行分类并授予不同的访问权限。③采取设置口令等方法, 当用户进入系统时进行安全性检查。④在日志中记录对数据的使用情况。⑤采用视图等方法对部分数据隐蔽。⑥数据加密等。

数据库系统的安全性可以从它的完整程度、灵活性、安全检查和额外开销以及安全机制自身的稳健性等若干方面来衡量。

(周立柱)

shuju bianma

数据编码 (data encoding) 一种把模拟数据或数字数据变成数字信号的转换技术。通过数据编码变换后的信号可通过数字通信介质传输出去。而在接收端将数字信号变成原来形式的变换被称为解码。

数字数据的数字信号编码

对于传输数字信号来说, 最普遍而且最容易的办法是用两个电压电平来表示两个二进制数字。例如, 无电压 (也就是无电流) 常用来表示 0, 而恒定的正电压用来表示 1。使用负电压 (高) 表示 1 也是很普遍的, 这种编码方法如图 1(a) 所示, 被称为不归零制 (NRZ)。

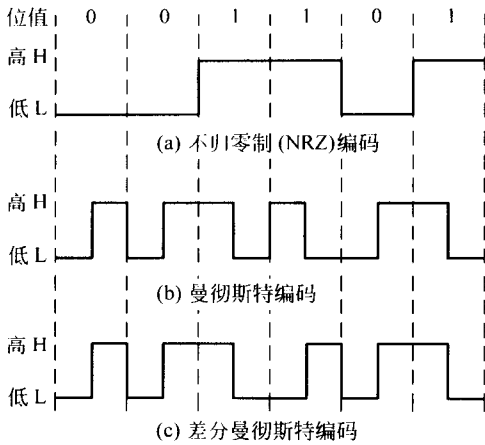


图1 数字信号的编码

不归零制的编码方法有若干缺点,因为它难以决定一位的结束和另一位开始,需要有某种方法来使发送器和接收器进行定时或同步。而且,如果传输中1或0占优势的话,那么电路在每位时间内将有累积的直流分量。这样,若使用变压器在数据通信设备和所处环境之间提供良好的绝缘的交流耦合是不可能的。此外,直流分量可使连接点产生电腐蚀或其他损坏。

克服上述缺点的另一个编码方案是曼彻斯特编码(见图1(b)),这种编码通常用于局部网络传输。在曼彻斯特编码方式中,每一位的中间有一个跳变。位中间的跳变既作为时钟,又作为数据;从高到低的跳变表示1,从低到高的跳变表示0。有时,人们也使用被称为差分曼彻斯特编码的修改格式(见图1(c))。在这种情况下,位中间的跳变仅提供时钟定时,用每位周期开始时有无跳变来表示0(1)的编码。在上述两种情况下,由于时钟和数据包含于信号数据流中,所以这种编码被人们称为自同步编码。

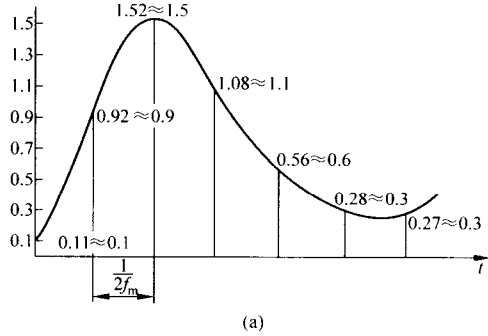
模拟数据的数字信号编码

利用数字信号来对模拟数据进行编码的最常见的例子是脉冲代码调制(PCM)。它常用于对声音信号进行编码。脉冲代码调制是以采样定理为基础的,采样定理指出:

如果在规则的时间间隔内,以高于两倍最高有效信号频率的速率对信号 $f(t)$ 进行采样的话,那么,这些采样值包含了原始信号的全部信息。利用低通滤波器可以从这些采样中重新构造出函数 $f(t)$ 。

如果声音数据限于4 000 Hz以下的频率,那么

每秒钟8 000次的采样可以满足完整的表示声音信号的特征。然而,值得注意的是,这只是模拟采样。为了转换成数字采样,必须给每一个模拟采样值指定一个二进制代码。图2表示了这样一个例子。图



(a)

数字	等效二进制数	脉冲代码波形
0	0000	
1	0001	
2	0010	
3	0011	
4	0100	
5	0101	
6	0110	
7	0111	
8	1000	
9	1001	
10	1010	
11	1011	
12	1100	
13	1101	
14	1110	
15	1111	

(b)

图2 脉冲代码调制

中每个采样值都被近似地量化为16个不同级中的一个,这样,每个采样值都能用4位二进制数来表示。由于采样值的量化级少,因此通过采样值来精确地恢复原始信号是不可能的。如果使用7位二进制表示采样的话,就允许有128个量化级,那么所恢复的声音信号的质量就可能达到模拟传输所达到的质量。这就意味着,仅仅是声音信号就需要有每秒钟8 000次采样 \times (每个采样)7位=56 000 b/s的数据传输速率。

人们通常使用称之为“非线性编码”的技术来改进脉冲代码调制方案。这种技术的特点是128个量化级不等分。等分的意义是不管信号的幅度大小,每个采样的绝对误差是同样的。因此,低幅值的地方相对容易变形。如果在低幅值处使用较多的量化级,而在较高幅值处使用较少的量化级,那么就可

使整个信号的变形显著减小。

参考文献

1. 胡道元. 计算机局域网(第三版). 北京: 清华大学出版社, 2002

2. Sheldon T. LAN Times Encyclopedia of Networking. McGraw-Hill Inc., 1994 (胡道元)

shuju cangku

数据仓库 (data warehouse) 面向主题的、集成的、数据不再更新的以及随时间不断变化的数据集。数据仓库用来存放海量的只读数据, 是为决策支持系统提供基础数据的分析型数据库。

数据仓库技术是 20 世纪 90 年代随着数据库应用由事务处理向分析处理扩展时兴起的计算机技术。传统的事务处理技术已经比较成熟, 它显著地提高了企业或部门事务处理的效率和水平。在实际运行过程中这些企业的数据库积累了大量的业务数据, 如产品数据、销售数据、客户数据及市场数据等。这些数据是宝贵的资源, 其中隐含着丰富的信息和有用的知识, 有可能对企业的决策产生重大影响。于是, 人们提出了利用数据库中的数据资源进行决策分析的新需求。

可以把数据处理划分为两大类: 操作型处理和分析型处理, 也称作联机事务处理 (OLTP) 和联机分析处理 (OLAP)。操作型处理也叫事务处理, 是指对数据库联机的日常操作, 通常涉及对数据库中当

前记录的查询和修改; 分析型处理是指基于数据的分析操作和辅助决策, 通常是对海量的历史的数据查询和分析。前者要求响应速度快、系统的并发度高和事务吞吐量高; 后者则访问的数据量非常大, 查询和分析操作复杂。两者之间的差异使得传统的数据库技术不能同时满足两类数据处理的要求, 数据仓库技术应运而生。

数据仓库本质上和数据库一样是长期储存在计算机内, 有组织、可共享的数据集合。数据仓库和数据库主要的区别是数据仓库中的数据具有以下四个基本特征: ①数据是面向主题的; ②数据是集成的; ③数据是不可更新的; ④数据是随时间不断变化的。

数据仓库中的数据组织具有不同的综合级别, 一般称之为“粒度”。粒度越大, 表示细节程度越低, 综合程度越高。

数据仓库系统总体上由以下几部分组成: 数据仓库的后台工具、数据仓库服务器、联机分析处理 (OLAP) 服务器和前台工具, 如图 1 所示。

数据仓库的后台工具, 包括数据提取、清洗、转换、装载和维护工具。它们负责对各种数据源进行转换, 然后装载到数据仓库中, 同时维护数据仓库和数据源数据的一致性。与数据仓库有关的系统目录很大, 需要在一个独立的数据库中存储与管理, 这个数据库称为元数据库。元数据 (参见多媒体数据库) 管理工具是数据仓库系统的一个重要组成部分。

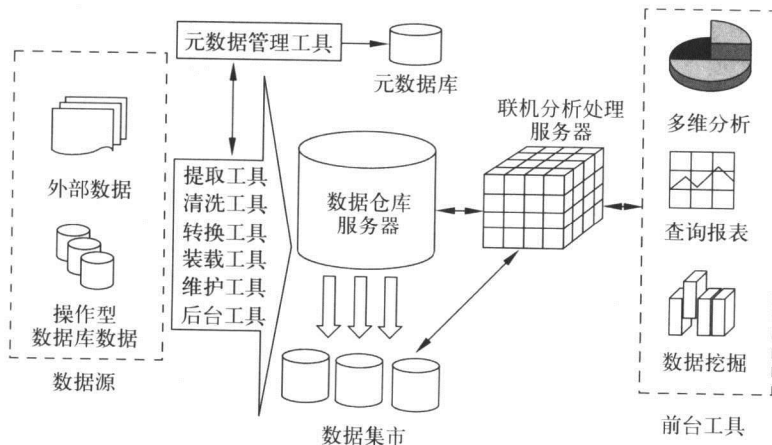


图 1 数据仓库系统的体系结构

数据仓库服务器负责储存和管理数据仓库的数据, 并给 OLAP 服务器和前台工具提供存取接口 (如

结构查询语言 SQL 查询接口)。数据仓库服务器目前一般是关系数据库管理系统 (RDBMS) 或扩展的

RDBMS。

数据仓库中存放着大量的数据,因此,其查询性能就成为对数据仓库技术的一大挑战。TPC-D 是对数据库中大量数据复杂查询的基准性能测试,可以用它来衡量和比较数据仓库的性能。数据集市是部门级的数据仓库。数据集市可以按业务来组织,也可以按主题或数据的地理分布来组织。

联机分析处理(OLAP)服务器透明地为前台工具和用户提供多维数据视图(参见**联机分析处理**)。

前台工具包括查询报表工具、多维分析工具、**数据挖掘(DM)**工具和分析结果形象化工具。

在整个系统中,数据仓库居于核心地位,是数据分析和挖掘的基础;数据仓库管理系统负责管理系统的运转,是整个系统的引擎;而数据仓库工具则是整个系统发挥作用的关键,只有通过高效的工具,数据仓库才能真正把数据转化为信息和知识,为企业和部门创造价值发挥作用。

数据仓库系统是一种解决方案,不是可以买到的现成产品。建立数据仓库系统是一项长期、艰巨和有风险的信息系统工程。

目前,数据仓库技术、联机分析处理技术和数据挖掘技术已经成为企业商务智能的三大技术支柱,并在金融、保险、税务、零售、航空及医疗等行业获得广泛应用。

参考文献

1. 萨师煊,王珊. 数据库系统概论(第三版). 北京:高等教育出版社,2000
2. 王珊等. 数据仓库技术和联机分析处理. 北京:科学出版社,1998
3. Inmon W H. Building the Data Warehouse. John Wiley & Sons Inc., 1996 (王珊)

shuju chuli sulü

数据处理速率 (processing data rate, PDR)

美国及巴黎统筹委员会用来限制计算机产品出口的系统性能指标估算方法,它按如下的公式来定义:

$$PDR = L/R$$

式中 L ——指令平均长度,即每条指令传送数据的平均位数,b;

R ——指令的平均执行时间, μs 。

$$L = 0.85G + 0.15H + 0.4J + 0.15K$$

$$R = 0.85M + 0.09N + 0.06P$$

式中 G ——定点指令字长($\geq 24b$),b;

H ——浮点指令字长($\geq 30b$),b;

J ——定点数长度,b;

K ——浮点数长度,b;

M ——定点加法时间, μs ;

N ——浮点加法时间, μs ;

P ——浮点乘法时间, μs 。

参加运算的两个数之一应在累加器或用作累加器的存储单元中,另一个在内存中,运算结果留在累加器或用作累加器的内存单元中。

从上面的公式可见,PDR 是根据定点、浮点加和乘法指令执行时间以及它们相关的数据宽度,按一定的混合比例对中央处理机与主存的数据处理速率进行计算的。它允许并行处理及先行取指令的功能,这时,所取的是平均指令执行时间。

带有高速缓存的计算机,因为存取速度加快其 PDR 值也就相应提高。

PDR 已在 1991 年 9 月停止使用,改为用 CTP (综合理论性能)计算。

参考文献

陈兴业. 计算机系统性能评价. 广州:华南工学院出版社,1987 (吴荣泉)

shuju chuanshu

数据传输(data transmission) 将源站的数据编码成信号,沿**传输介质**传播至目的站的过程和技术。数据传输的品质取决于被传输信号的品质和传输介质的特性。被传输的信号既是时间的函数,也可被表示成频率的函数,即信号由不同的频率成分组成。传输的数据可分为模拟数据和数字数据两种,**模拟数据**是连续的值,如语音和视频;**数字数据**是离散的值,如文本信息和整数。

时域和频域

时域概念 从时域概念分析,信号是时间的函数,可分成连续信号和离散信号两种。如果相同的信号形式能周期性地重复,则称为“周期信号”,其数学表达式为:

$$S(t+T) = S(t) \quad -\infty < t < +\infty$$

式中 T 为信号周期。

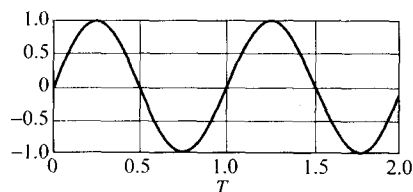
正弦波是基本的连续信号,它可用三个参量表示,即幅度(A)、频率(f)和相位(ϕ)。因此正弦波可表示为:

$$S(t) = A \sin(2\pi ft + \phi)$$

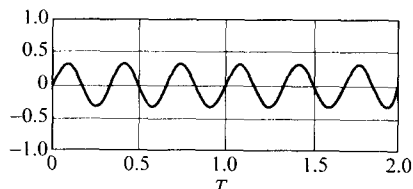
频域概念 从频域概念分析,信号可由多个频率成分组成,可表示为:

$$S(t) = \sin(2\pi f_1 t) + 1/3 \sin[2\pi(3f_1)t]$$

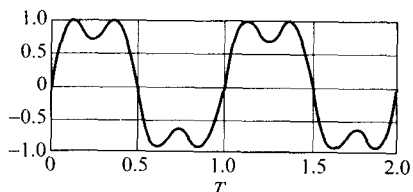
如图 1 所示。



(a) $\sin(2\pi f_1 t)$



(b) $1/3 \sin[2\pi(3f_1)t]$



(c) $\sin(2\pi f_1 t) + 1/3 \sin[2\pi(3f_1)t]$

图 1 多个频率成分组成的信号

信号 $S(t)$ 是由频率 f_1 和频率 $3f_1$ 合成,后者是前者的整数倍。频率 f_1 是合成信号的基本频率成分。信号周期也等于基本频率信号的周期,即都等于 T 。

一个信号的**频谱**指它包含的频率范围。一个信号的**频宽**指它的频谱的宽度。大部分的信号能量包含在相对窄的频带内,这个频带称**有效频带**或**频带**。

虽然一个信号的波形可能覆盖很宽的频带,但实际上,任何传输介质只能适应于有限的频带,也就是说在传输介质上能传输的数据速率是有限制的。数据速率和频宽有直接关系,数据速率越高,有效频带就越宽。传输系统的频带越宽,系统能传输的数据速率就越高。如果某数字信号数据速率为 W ,则 $2W$ Hz 频带的系统就能很好地满足该信号的传输要求。

模拟数据传输和数字数据传输

在通信系统中,利用电信号把数据从一个点传到另一个点。**模拟信号**是一种连续变化的电磁波,这种电磁波可以按照不同频率在各种介质上传输;**数字信号**是一系列的电压脉冲,用恒定的正电压来表示二进制 1,用恒定的负电压来表示二进制 0。

用数字信号发送数据的最基本的优点是比用模拟信号发送数据便宜,而且很少受噪音干扰的影响。最主要的缺点是数字信号比模拟信号易衰减。

模拟数据和数字数据都可以用模拟信号或数字信号来表示,因而也可以用这些形式传播,图 2 描述了这种情况。一般来说,模拟数据是时间的函数,并且占有一定的频谱范围。这种数据可以直接用占有相同频谱范围的电磁波信号来表示,最好的例子是声音数据。作为声波,声音数据的频率范围在 20 Hz ~ 20 kHz 之间。然而,大多数语音(言语)能量的范围要窄得多。声音信号的标准频谱是 300 ~ 3 400 Hz。对于清楚地传播声音来说,这个频谱是完全足够的,电话设备恰恰是这样做的。为了将所有的声音以 300 ~ 3 400 Hz 输入到通信系统中,需要产生有相同频率幅度的电磁信号。也可以用相反的过程把电磁能转换为原来的声音。

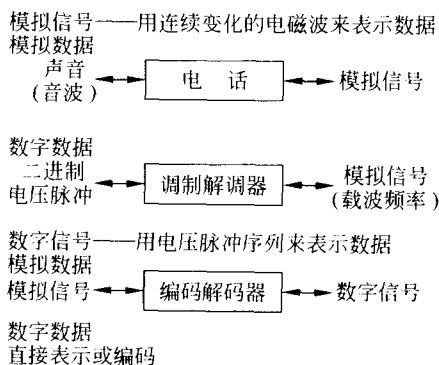


图 2 模拟和数字数据的模拟和数字信号

利用**调制解调器**,数字数据也可以用模拟信号来表示。调制解调器通过一个载波频率把一串二进制(双值)电压脉冲转换为模拟信号。所产生的信号占有以此种载波频率为中心的某一频谱,并且能在适合于此种载波的介质上传播。大多数通用的调制解调器都用音频频谱来表示数字数据,因此能使那些数据在普通的音频电话线上传播。在线路的另一端,调制解调器把信号解调为原来的数据。

与调制解调器完成的操作相类似,模拟数据也可以用数字信号来表示,对于声音数据来说,完成这种功能的是**编码解码器**。**编码解码器**接收一个直接表示声音数据的模拟信号,然后用二进制位流近似地表示这个信号。在线路的另一端,二进制位流被重新构造为模拟数据。

数字数据也可以直接用两种电平来表示,即用

二进制形式表示。然而,为了改变其传播特性,常常对二进制数据进行编码。

模拟信号和数字信号都可以在合适的传输介质上进行传输,但模拟信号和数字信号之间终究还是有差别的。

模拟传输是一种不考虑内容的传输模拟信号的方法,信号可以表示模拟数据(例如声音)或表示数字数据(例如通过调制解调器发送的数据)。无论是哪种情况,模拟信号在传输一定的距离之后都将衰减。为了实现长距离传输,模拟传输系统都包括放大器,用放大器来使信号中的能量得到增加。但是放大器也使噪音分量增加。如果通过串联放大器来实现长距离传输,那么信号就会越来越畸变。对于模拟数据(例如声音)可以允许许多位的变形,而且仍然易于理解。但是,对于数字数据来说,串联的放大器产生的畸变将会导致错误发生。

与上述模拟传输相反,数字传输与信号的内容有关。衰减会危及数据的完整性,因此数字信号只能在一个有限的距离内传输。为了获得更大的传输距离,可以使用中继器。中继器接收数字信号,把数字信号恢复为1和0的模式,然后重新传输这种新的信号。这样,就克服了衰减。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999

2. Stallings W. Data and Computer Communications. Prentice Hall, 1997

(胡道元)

shuju chuanshu chacao jiance yu kongzhi
数据传输差错检测与控制 (error detection and control of data transmission) 在传输过程中对可能发生错误的帧进行检测和纠正的控制机制。常用的差错检测方法有奇偶检验和循环冗余检错两种。差错控制方法有停-等自动重复请求 (ARQ)、返回 N 帧的自动重复请求和选择性重发、自动重复请求等多种方法。

差错检测

奇偶检验 这是一种最简单的检错方法。例如,在传输 ASCII 字符时,每个 ASCII 字符用 7 位表示,最后加上一个奇偶位总共成为 8 位。对于偶检验来说,最后加上的奇偶位使整个 8 位中的 1 的个数为偶数。对于奇检验来说,则整个 8 位中的 1 的个数为奇数。例如发送 ASCII 字符 G(1110001),采用奇检验时,奇偶位为 1,即传输 11100011。接收器

检查接收到的数据的 1 的个数为奇数,就认为无错误发生。采用奇偶检验时,若其中两位同时发生错误,则会发生检测不出错误的情况。因此对于高数据速率或者噪声持续时间较长的情况,由于可能发生多位出错,奇偶检验就不适用了。偶检验一般用于同步传输,而奇检验一般用于异步传送。

循环冗余检错 循环冗余检错 (CRC) 是一种最普遍、最有效的检错方式。它的基本原理如下:已知一个数据块包含有 k 位,发送器产生一个包含有 n 位的序列,这个序列称为帧检验序列 (FCS),把 FCS 加到数据块的后面组成一个包含有 $k+n$ 位的发送帧 T ,使得 T 能够用一个已知数 P 整除。接收器把接收到的帧除以 P ,若没有余数,则认为没有出错。下面列出两种 CRC 的算法。

(1) 模 2 算法

假设: M 是包含有 k 位的一个数据块;

F 是包含有 n 位的 FCS, $n < k$;

P 是已知的除数,等于 $n+1$ 位;

T 是发送帧,包含 $k+n$ 位。

模 2 算法的过程是:用 $2^n M$ 除以 P ,得到的余数 R 即是 FCS。注意,在模 2 算法中,加减法只作异或操作,没有进位。如果接收到包含有位错的帧也能用 P 整除,传输中的位错就不能检测到,但这种情况发生的概率是极小的。

(2) 多项式算法

第二种算法是多项式算法。把二进制用一个伪变量为 X 的多项式表示。每一项的系数由二进制的值来决定。例如对于 $M=110011$,可用 $M(x) = x^5 + x^4 + x + 1$ 表示。

$P=11001$ 可用 $P(x) = x^4 + x^3 + 1$ 表示。算术运算仍用模 2 算法。则 CRC 过程可用下式表示:

$$x^n M(x) / P(x) = Q(x) + R(x) / P(x)$$

$$T(x) = x^n M(x) + R(x)$$

如果接收到的帧包含有位错,但能被 $P(x)$ 整除,那么所发生的位错就没有被检测出来。选择恰当 $P(x)$ 可以减少这种可能性,常用的 $P(x)$ 有如下几种:

$$\text{CRC-16} = x^{16} + x^{15} + x^2 + 1$$

$$\text{CRC-CCITT} = x^{16} + x^{12} + x^5 + 1$$

$$\text{CRC-32} = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

差错控制

差错控制涉及以下几个方面:①差错检测;②正响应,即目的站对于成功到达且无错误的帧发

回一个正响应；③超时重传，即对于在预定时间内没有发回响应的帧，发送站要重新发送这一帧；④负响应并重传，即目的站对出错的帧发回一个负响应，发送站重新传送这些帧。

以上几方面概括起来可称为自动重复请求 (ARQ)，采用 ARQ 后，使原本不可靠的数据链路变成可靠。ARQ 大致有下列三种标准：①停-等 ARQ；②退回 nARQ；③选择重发 ARQ。以上三种机制都基于流控技术。

停-等 ARQ 基于停-等流控技术，其原理如图 1 所示。在帧的传输过程中可能发生以下两种错误：第一种是接收站用检错技术断定接收到的帧有错并将它丢弃。发送站发送帧后启动一个定时器，如果在给定的时间内没有接收到响应，则重新发送此帧。当然发送站应持有发送帧的副本。第二种是响应遭到破坏。假设 A 发送帧后，B 站接收到该帧，经检验断定为接收到的是正确的帧并发送一个响应，但在传送过程中响应遭破坏以致 A 没有在预定时间内接收到响应，A 就重新发送此帧，造成 B 先后接收到相同的两帧。为避免上述情况，帧都用 0 或 1 交替编号。正响应用 ACK0 和 ACK1 表示。ACK0 表示已接收到帧 1，并准备接收帧 0。

停-等 ARQ 的优点是非常简单，其缺点是传输效率低。

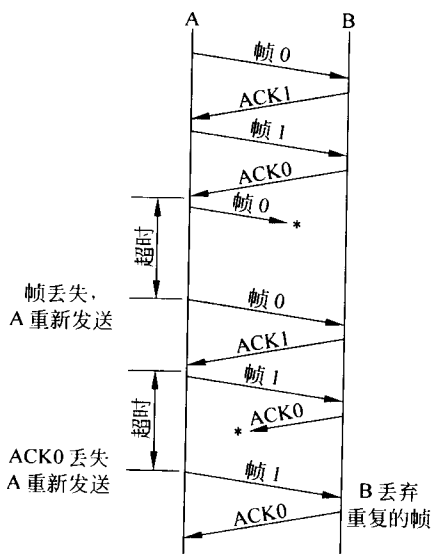


图 1 停-等 ARQ 差错控制

退回 nARQ 基于滑动窗口流控技术，其原理

如下：

A, B 两站分别用滑动窗口流控技术进行数据交换。若没有错误发生，目的站发回一个 RR 响应。若目的站检测到有错的帧，则发送一个负响应 REJ，并丢弃这一帧及其后接收到的所有帧，直到重新接收到正确的帧。因而发送站在接收到 REJ 后，必须重传发生错误的帧以及先前已发送过的其后的帧。

图 2 是一个退回 nARQ 的例子。由于链路的传播延迟，在响应接收到之前，发送站又发送 2 帧。因此当接收到 REJ5，就不但要重传帧 5，而且要重传帧 6，帧 7。因此发送器要持有所有没有得到响应的帧的副本。

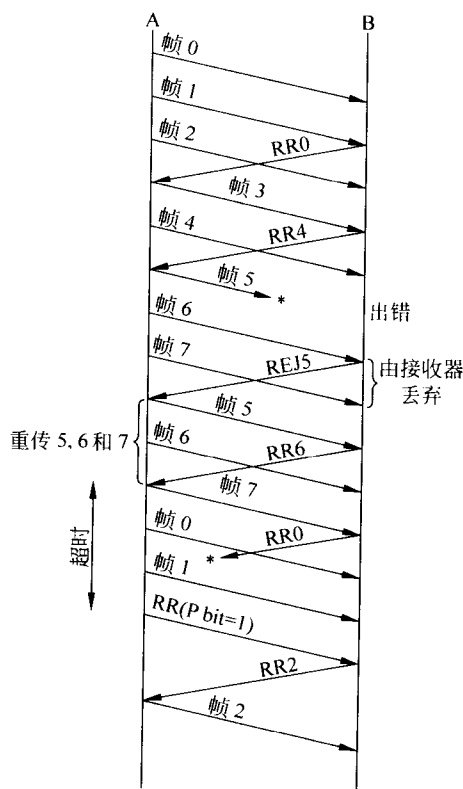


图 2 退回 nARQ 差错控制

应用滑动窗口流控时，序号若占用 k 位，则序号的范围是 2^k ，但最大的窗口应限于 2^{k-1} ，否则会引入问题。例如序号用 3 位，则帧的序号从 0 至 7。假设站发送帧 0 并得到一个响应 RR1，然后发送帧 1、2、3、4、5、6、7、0 并得到响应 RR1。这既可以认为 8 帧都已正确地收到，也可以理解为 8 帧都遭破坏或者丢失。如果把窗口限于 $2^3 - 1 = 7$ ，则可

避免上述问题的发生。

选择性重发 ARQ 当接收到某一帧的负响应时,只需重传该帧,这种机制称为 SR E J。它比退回 n ARQ 的效率高。但接收器必须要有足够的缓冲器来存放那些没有出错的帧直到接收到重传的帧,并且必须要有一定的逻辑功能,以便把重传的帧插入到正确的位置。发送器也需要比较复杂的逻辑功能使得可以不按顺序号发送帧。因此选择性重发 ARQ 比退回 n ARQ 较少使用。对于滑动窗口的范围的限制也比退回 n ARQ 更严。窗口的最大范围应定为 2^{k-1} , 对于 3 位顺序号, 则窗口范围为 4。

参考文献

1. 胡道元. 计算机网络(高级). 北京: 清华大学出版社, 1999

2. Stallings W. Data and Computer Communications. Prentice Hall, 1997 (胡道元)

shuju chuansong

数据传送 (data transfer) 数据在计算机内部或计算机之间或计算机系统与其他系统之间的传送操作。

数据传送可以分为 3 个层次: ①数据从一个计算机系统传送到另一个计算机系统; ②在计算机系统中, 数据从一个外围设备(或存储单元)传送到另一个外围设备(或存储单元); ③数据在中央处理机内部各寄存器之间的传送。数据在传送的过程中, 不应改变其格式和内容。

衡量数据传送的性能指标主要有: ①数据传送速度: 在数据通道上传送数据的速度, 称为**数据传送率**, 也称数据传输速率, 一般以每秒传送的字节、千字节或兆字节的数量作为度量的单位, 记为 b/s, kb/s 或 Mb/s。在有的情况下, 也用每秒传送二进制信息位的数量作度量单位, 记为 b/s, kb/s 或 Mb/s。有时也用**平均数据传送率**(或称**有效数据传送率**)来度量, 这是指在一段时间内, 由一个外围设备(或存储单元)到另一个外围设备(或存储单元)传送数据的速率。这段时间包括数据块、字或记录之间的间隔时间, 但不包括程序启动、查找和停止等所用的时间。②**数据宽度**: 指每次并行传送的数据的位数, 例如 8 位, 16 位, 32 位, 64 位。

数据传送有多种不同的分类方法。按数据传送的方式, 可分为: ①**并行传送**: 在并行传送通路上, 输出数据的所有信息位并行地同时被传送到目的设备。并行传送的速度较快, 但所用的器件较多。

②**串行传送**: 在串行传送的通路上, 输出数据中的各信息位依次一位一位地传送。串行传送的速度较慢, 但逻辑实现较简单, 所用的器件较少。按传送双方是否采用统一的时钟, 数据传送可分为: ①**同步数据传送**: 传送双方采用统一的时钟脉冲。同步传送的优点是传输速率高, 缺点是受干扰后易引起同步错。②**异步数据传送**: 传送双方不采用统一的时钟脉冲, 发送和接收设备各按自己的时钟频率工作。按数据传送的控制方法, 可分为: ①**无条件传送**: 不查询接收设备是否处于就绪状态而强迫执行操作。为使传送可靠, 程序员必须了解接收设备的动态工作情况。②**条件传送**: 满足条件时才进行的数据传送。为此, 控制程序必须设有测试接收设备工作状态的指令。③**程序中断传送**: 由程序控制, 采用中断方式进行的数据传送。当程序运行到某一时刻, 由程序控制启动指定的外围设备, 然后主机继续执行原来程序。外围设备做好数据传送准备后, 向中央处理机发出中断请求信号, 中央处理机经过中断处理后, 停止正在运行的程序, 转向执行传送指令, 以完成主机和外围设备之间的数据传送, 传送完毕后, 再返回执行被中断的程序。④**直接存储器存取(DMA)**: 外围设备与存储器之间不需要中央处理机干预而直接由 DMA 控制器来控制的高速数据传送。(孙德文)

shuju dianlu shebei

数据电路设备 (data circuit equipment, DCE) 位于**数据终端设备(DTE)**和传输线路之间的中介通信设备。DTE 只有有限的数据传输能力, 很少直接接到传输线路或网络设施上, 而是通过 DCE 作为中介设施连接。DCE 负责在传输介质上发送和接收信号。在传输介质上交换信号的两个数据电路设备之间, 以及数据终端设备和数据电路设备之间必须遵守相同的规程。

数据电路设备提供了数据终端设备与通信网络的互连, 且接收数据和提供线路的时钟同步。当通信介质是传送模拟信号的电话线时, 用**调制解调器**作数据电路设备。当线路传送数字信号时, 用**信道服务部件(CSU)**和**数据服务部件(DSU)**作数据电路设备。

DTE 和 DCE 接口由开放系统互连基准(参考)模型的物理层规定, 最通用的标准是电子工业协会(EIA)RS232C 和 RS232D。其他还有 V. 24, RS449, V. 35, X. 21 等标准。DTE 和 DCE 设备一般采用 25

芯连接器进行连接。

参考文献

Sheldon T. LAN Times Encyclopedia of Networking. McGraw-Hill Inc., 1994 (胡道元)

shuju tiaozhi

数据调制 (data modulation) 一种把模拟数据或数字数据变成模拟信号的转换技术。调制是载波信号的某些特性根据输入信号而变化的过程。所有调制技术涉及到载波信号的下列一个或几个参数的变化: 幅度、频率和相位。无论是模拟数据还是数字数据, 原始输入数据经过调制就可作为模拟信号通过通信介质发送出去, 并将在接收端进行解调, 再变换成原来的形式。

模拟数据的模拟信号调制

模拟数据经由模拟信号传输时不需进行变换, 但是, 由于考虑到天线发送时的天线尺寸问题, 模拟形式的输入数据也需要在甚高频下进行调制。其输出信号是一种带有输入数据的频率极高的模拟信号。有三种不同的调制技术: 调幅、调频与调相, 其中最常用的两个调制技术是幅度调制 (AM) 和频率调制 (FM)。

幅度调制是一种使载波的幅度随着原始模拟数据的幅度变化而变化的技术。载波的幅度会在整个调制过程中变动, 而载波的频率是相同的。接收端对接收到的幅度调制的信号进行解调, 以恢复成原始的模拟数据。

频率调制是一种使高频载波的频率随着原始模拟信号的幅度变化而变化的技术。因此, 载波频率会在整个调制过程中波动, 而载波的幅度是相同的。接收端对接收到的频率调制的信号进行解调, 以恢复成原始的模拟数据。

相位调制是一种使载波的相位随着原始模拟数据的幅度变化而变化的技术。载波的相位会在整个调制过程中变动, 而载波的幅度是相同的。接收端对接收到的相位调制的信号进行解调, 以恢复成原始的模拟数据。

数字数据的模拟信号调制

数字数据的模拟信号调制是通过改变载波的下列特性之一来实现的: 振幅、频率、相位, 或者是这些特性的某种组合。图 1 给出了对数字数据的模拟信号进行调制的三种基本形式: 幅移键控法 (ASK)、频移键控法 (FSK)、相移键控法 (PSK)。

(1) 在**幅移键控法 (ASK)**方式下, 用载波频率

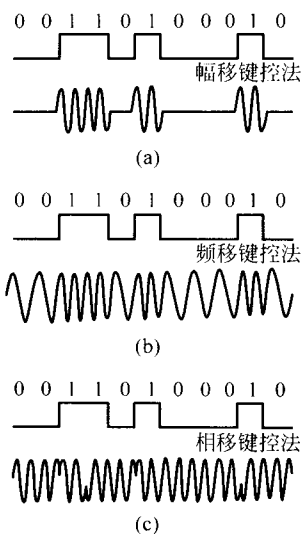


图 1 数字数据的模拟信号调制

的两个不同的振幅来表示两个二进制值。在有些情况下, 一个振幅为零, 即用振幅恒定的载波的存在来表示一个二进制数字; 而另一个二进制数字用载波的不存在表示。ASK 方式容易受设备增益变化的影响, 因此, 是一种低效率的调制技术。在音频线路上, 通常只能达到 1 200 b/s。

(2) 在**频移键控法 (FSK)**方式下, 用载波频率附近的两个不同频率来表示两个二进制值。这种方案与 ASK 方式相比, 不容易受干扰的影响。在音频线路上, 通常可达 1 200 b/s。这种方式一般也用于高频 (3 ~ 30 Mb/s) 的无线电传输, 它甚至也能用于较高频率使用同轴电缆的局部网络。

图 2 表示在音频线路上使用频移键控法进行全双工传输操作的例子。全双工传输指的是, 可以同时两个方向传输数据。为了实现这个目的, 一个带宽用于发送, 而另一个带宽用于接收。在一个方向 (发送或接收) 上, 调制解调器可以通过 300 ~ 1 700 Hz 频率范围内的信号。用来表示 1 和 0 的两个频率以 1 170 Hz 为中心, 两边各有 100 Hz 的移位。与此类似, 对于另一个方向 (接收或发送) 来说, 调制解调器可以通过频率为 1 700 ~ 3 000 Hz 的信号, 并且使用 2 125 Hz 为中心频率。在每对频率周围的阴影区指出了每个信号的实际带宽。值得注意的是, 几乎不存在交叠, 因此也几乎没有干扰。

(3) 在**相移键控法 (PSK)**方式下, 用载波信号的相位移来表示数据。图 1 中 (c) 表示的是一个

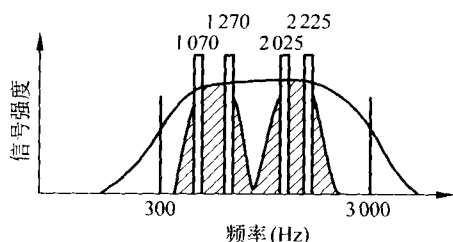


图2 频移键控法 FSK 传输

两相系统的例子。在这个系统中,0 被表示为发送与以前所发送信号串同相的信号串。1 被表示为发送与以前发送信号串反相的信号串。相移键控法也可以使用多于两相的位移。四相系统能把每个信号串编码为两位。PSK 技术有较强的抗干扰能力,而且比 FSK 方式更有效;在音频线路上,传输速率可达 9 600 b/s。

上述各种技术也可以组合起来使用。比较常见的是相移键控法和幅移键控法的组合,组合后在两个振幅上均可以分别出现部分相移或整体相移。

参考文献

1. 胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002

2. Sheldon T. LAN Times Encyclopedia of Networking. McGraw-Hill Inc., 1994 (胡道元)

shuju gongxiang

数据共享 (data sharing) 为提高数据资源的利用率,由多个用户(程序)按一定规则共同享用数据库中数据的一种技术。

数据共享是数据库和数据库以前的文件系统的重要区别之一。在数据库出现以前,数据处理以程序为中心,每一程序都有各自所用的数据文件。同一数据为多个程序使用时要重复存放多份,而数据库的数据为有关用户公用,集中管理所需的全部数据。数据库的数据共享有下列好处:①减少重复存放,节省存储器资源,②简化了对共享数据的修改,③保证了数据一致性。

为了实现数据共享,数据库管理系统要解决好两个问题:一是安全控制,即保证数据库中的数据不能被无权使用的用户读取或修改;二是并发控制,即处理各个程序同时工作时的相互干扰,如协调好两个程序同时修改同一数据而产生的冲突。

在以网络技术为基础的分布式数据库中,为了

减少网络中的通信数量,常常有控制地在网络不同节点上存放某些数据的复本。保证系统各复本的一致性,也是并发控制的一项任务。

参考文献

Ullman J D. Principles of Database Systems (Second Edition). London: Pitman Publishing Ltd., 1987

(楼荣生 朱扬勇)

shuju jiegou

数据结构 (data structures) 由若干数据成分按照一定方式构成的复合数据以及作用于其上的函数或运算。数据成分及其间的数据约束关系合称为数据结构的逻辑构成或逻辑结构。数据结构从数学上可以用适当的数学结构以及在其上的函数变换统一地定义。

今以字符串为例,它的逻辑结构是一组同一类型(字符型)的字符,以及在该字符串中字符间的前后顺序关系。在此结构上的常见运算有:求字符串的长度、将字符串分为两部分(首字符及其余部分)以及求两字符串拼接后的新串等等。在设计数据结构阶段,应该视应用需要确定适当的运算。这对于设计字符串数据结构是很重要的。

一个数据结构的存储结构是指它在计算机中所需用的存储空间、空间的构成结构及对该存储结构的访问方式等的统称。一般来说,存储结构的具体设计是在明确该数据结构的定义之后进行的。

在传统的程序设计语言(如 PASCAL, C 等)中,定义数据结构的基本途径是采用数据类型。简单的数据结构是用单一的标准数据类型,如整型、字符型、布尔型或实型等所定义的。在使用时,程序对其采取整体性访问方式,即读写访问只对整体而不涉及对其某构成部分的访问(如其中某二进制的内容)。由若干较简单的数据结构,运用程序语言所提供的复合构成方法,诸如数组、记录、集合等,可以构造更为复杂的数据结构。复杂数据结构的特点是:对它既可进行整体性访问,也可单独对某构成成分进行访问。以上所述,虽已被多数熟知的程序设计语言所采用,但是有其局限性。它偏重于逻辑结构及存储结构的构成方面,而对其运算等语义部分的描述不足。根据抽象数据类型理论,程序应将数据结构的逻辑构成和它的运算在一起定义,并封装成一个整体。而且数据结构的具体实现,包括采用的存储结构和运算操作的具体算法实现,应该与上述封装相分离,分别地加以描述。新近使用的程

序语言,如 Ada 以及面向对象语言(如 C++, 或 Smalltalk 等)皆采用了这一思想。

存储结构涉及存储分配和数据访问两个方面。鉴于计算机的主存储器具有随机访问及相邻地址顺序存储等物理特点,常把经常被同时访问的数据安排在相邻的物理存储区域内,这称为顺序存储结构。一维数组、字符串和记录等定长的数据结构多采取这种结构,并运用静态存储分配方式:即在程序编译连接时(程序运行前)确定所占用的存储区域的物理地址范围。对顺序存储结构而言,常用的数据访问方式有顺序访问、索引访问及散列访问等多种。另一种存储结构是链接结构。与顺序存储结构不同,它常常采用动态存储分配方式:在程序运行中当需要创建新的数据结点时刻,申请和分配存储空间;当删除数据结点时则及时释放其占用空间。这种分配方式适合于变长度的数据结构,如具有递归逻辑关系的结构,如表、树等情形。

数据的逻辑结构从比较抽象的角度——与存储结构相对独立,刻画数据结构所具有的数学性质:将数据元素抽象为结点,数据元素间的关系当作连接结点的边,而访问数据结构的运算则描述为离散数学结构上的运算。常见的逻辑结构有线性结构、树结构、图结构等。

参考文献

Wirth N. Algorithms + Data Structures = Programs. London: Prentice-Hall Inc., 1976 (许卓群)

shuju jiegouhua xitong kaifa fangfa

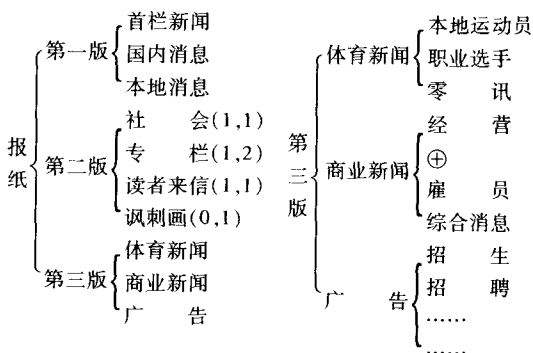
数据结构化系统开发方法 (data structured system development method) 一种面向数据结构的软件开发方法。又称 Warnier-Orr 方法。该方法利用 Warnier 图表示数据的层次结构,并且依据一定的步骤把层次的数据结构映射到程序结构。

1974 年法国计算机科学家 J. D. Warnier 提出了一种表示信息层次结构的图形工具——Warnier 图,他利用三种结构成分,即顺序、选择、重复来表示信息的层次结构,进而导出软件的结构。80 年代初, K. Orr 将其扩充,形成了数据结构化系统开发 (DSSD) 方法。该方法既考虑了数据流和功能特性,也考虑了数据的层次结构。

Warnier 图 是利用树形结构表示信息层次结构的一种紧凑而直观的图形工具。用 Warnier 图可以表明信息的逻辑组织,它可以指出一类信息是重

复出现的,也可以表示特定信息在某一类信息中是有条件地出现的。因为重复和条件约束是说明软件处理过程的基础,所以很容易把 Warnier 图转变成软件设计的工具。

图 1 以报纸自动编辑系统为例,说明这种图形工具的用法。图(a)是报纸各版的构成,图(b)则是第三版的详细栏目。花括号表示层次关系,括号内从上到下是顺序的信息项;异或符号“ \oplus ”表明一类信息或一个数据元素在一定条件下才出现,而且在这个符号上、下方的两个名字只能出现一个;一个名字右边的圆括号中的符号指明了这个信息类(或元素)在这个数据结构中重复出现的次数。



(a) Warnier 图

(b) 第三版细目图

图 1 报纸自动编辑系统 Warnier 图

数据结构化系统开发方法把系统开发分为分析和设计两个阶段。

在分析过程中,首先研究应用环境,即从信息的产生者和接收者的观点来观察数据是如何在两者之间流动的。然后,用类似 Warnier 图的层次结构表示方法来表达应用的功能。最后,再用 Warnier 图来描述应用结果。

这一方法在需求分析时会涉及到信息域的所有属性:信息流、信息内容和信息结构,其分析步骤包括:

(1) 确定应用环境 为了确定问题的应用环境,对问题作出描述,需要回答以下三个问题:要处理的信息项有哪些?谁是信息的产生者和接收者?每个信息的产生者或接收者都与环境中的哪些信息项有关? DSSD 方法以实体图作为工具回答这些问题。

(2) 表达应用功能 研究跨越系统边界的信息流可以弄清楚所开发系统应该实现的功能。使用类

似 Warnier 图的层次结构表示法,可以把信息和加工联系起来,称之为作业线图。在该图中,每个信息流项是前面编号信息项与产生所需项的过程结合起来得到的。每个加工过程由一个处理说明细化。该说明包括输出、动作、动作的频率及输入。

(3) 给出应用结果 DSSD 方法要求对系统的输出建立雏型,以表明主要的系统输出及构成输出的信息项组织。有了这一雏型就可利用 Warnier-Orr 图来模拟信息的层次结构。

DSSD 方法设计过程以分析过程的需求规约作为设计的根据,需求规约包括应用环境、功能描述及应用结果。所有表述这些信息的数据和图表是 DSSD 设计的基础。设计过程着眼于系统输出、界面设计和软件的过程设计,它以输出数据结构(称逻辑输出结构)和对对应过程(称逻辑处理结构)的详细描述作为设计过程的结果。

DSSD 是一种公式化和综合性的设计方法,其设计步骤为两部分:

(1) 逻辑输出结构(LOS)的获得 LOS 的获得可以通过以下 4 个步骤:①对问题进行描述并对有关信息进行量化,将所有可区别的不可再分的数据项(称为原子数据项)一一列出。②指明各原子数据项出现的频率。③对可被再分的数据项进行量化。④作出 LOS 的图表表示。

逻辑输出结构是信息项的分级表示,其中的信息项由基于计算机的系统输出构成。提取 LOS 的第一步是分离所有的原子数据项,这可由对问题陈述的了解,或审查映象报告的格式来取得。其次,各原子数据的出现频率应明确地指出。当定义了所有的原子数据项并列出其出现频率后,设计者再查出具有全局性质的信息项,即所谓全局信息项。

(2) 逻辑处理结构(LPS)的提取 逻辑处理结构(LPS)处理对应 LOS 所需软件的过程表示。DSSD 方法中提取 LPS 与早期 Warnier 方法提取程序逻辑构造(LCP)有相似之处。每一个全局数据项都成为不断加入过程指令的重复结构。提取 LPS 的步骤如下:①从表示 LOS 的 Warnier-Orr 图中提出所有原子数据项。②将 BEGIN 和 END 作为分界符加入所有全局数据项。③定义所有初始和结束的指令或进程。④指出所有可计算的和非数字的进程。⑤指明所有输出指令和进程。⑥指明所有输入指令和进程。

DSSD 方法是一种系统的软件分析和设计方

法。由于这种开发方法提供了具体的工作步骤以取得详细的过程设计,所以为自动生成源代码提供了可能。由此方法形成的自动代码生成系统可通过对逻辑和实体输出及过程结构的精确描述来生成所需代码。

参考文献

1. Warnier J D. Logical Construction of Programs. Van Nostrand Reinhold, 1974
2. Orr K T. Structured Systems Development. New York: Yourdon Press, 1977 (郝克刚 葛玮)

shujuk

数据库(database) 长期储存在计算机内、有组织、可共享的数据集合。数据库中的数据按一定的数据模型组织、描述和储存,具有较小的冗余度,较高的数据独立性和易扩展性,并可为各种用户共享。整个数据库在建立、运用和维护时由**数据库管理系统(DBMS)**统一管理、统一控制。用户能方便地定义数据和操纵数据,并保证数据的安全性、完整性、多用户对数据的并发使用及发生故障后的数据库恢复。数据库是数据库系统的一个重要组成部分。

数据库在历史上曾出现过三个词: databank, data base 和 database。

databank 出现在 20 世纪 60 年代后期的一个数据系统中。当时, databank 储存的主要是数字和符号,并通常用于股票、债券等领域。以后, databank 一词逐渐被 database 所取代。

data base 出现在 20 世纪 60 年代初,顾名思义是存放大量数据的“仓库”。

database 出现在 1964 年数据处理方面的文献,最早是美国军方人事部门采用。60 年代后期 database 一词为整个数据处理领域广泛采用。

现在, data base 和 database 可以混用,而 database 的使用更为普遍。

数据库按数据模型分,可分为**层次数据库**、**网状数据库**、**关系数据库**和**面向对象数据库**。

数据库技术与其他学科的技术内容结合,出现了各种新型数据库:

数据库技术与分布处理技术结合,出现了**分布式数据库**;

数据库技术与并行处理技术结合,出现了**并行数据库**;

数据库技术与人工智能结合,出现了**演绎数据**

库和知识库;

数据库技术与多媒体技术结合,出现了多媒体数据库;

数据库技术用于特定的领域,出现了工程数据库、地理数据库、统计数据库、空间数据库等特定领域数据库。

数据库中的数据有逻辑和物理两个侧面。数据的逻辑结构的描述称为**逻辑模式**,逻辑模式分为描述全局逻辑结构的全局模式(简称模式)和描述某些应用的局部逻辑结构的子模式(外模式)。数据的物理结构的描述称为**储存模式**(内模式)。这三个模式是对数据的三个级别的抽象。数据库同时提供了子模式与模式之间、模式与储存模式之间的映射,从而保证了数据库中的数据具有较高的物理独立性和一定的逻辑独立性。

数据库不仅储存实体,而且体现实体之间的联系。数据库中的数据包括:数据本身、数据描述(即对数据外模式、模式、内模式的描述)、数据之间的联系和数据的存取路径。数据库中的数据是整体结构化的。数据不再面向某一程序,从而大大减小了数据冗余度和数据之间的一致性。同时,对数据库的应用可以建立在整体数据的不同子集上,使系统易于扩充。

数据库的使用者大致可分为三类:最终用户、数据库应用系统开发人员和数据库管理员 DBA。数据库最终用户指通过应用系统的用户界面使用数据库的人员,他们一般是业务人员。数据库应用系统开发人员包括系统分析员、系统设计员和程序员。系统分析员负责应用系统的分析,他们和用户、数据库管理员相结合,参与数据库设计;系统设计员负责应用系统设计和数据库设计;程序员则根据设计要求进行编码。数据库管理员是数据管理机构的一组人员,他们是拥有最高特权的数据库用户,负责全面管理数据库系统。

数据库在对大量信息的有效储存和快速存取方面发挥了重要作用,成为大型信息系统的核心和基础。数据库的整体概念、技术内容、应用领域,甚至某些原理均有重大的发展。数据库中数据的类型由传统意义的数字、字符发展到正文、声音、图形、图像等多种类型。应用领域从传统的面向商业与事务处理扩展到科技、经济、社会、生活的各个领域。

参考文献

1. 萨师煊,王珊. 数据库系统概论(第二版). 北京:高等教育出版社,1991

2. Date C J. An Introduction to Database System. 6th Edition. Reading: Addison - Wesley Publishing Company, 1995 (萨师煊 王珊)

shujuku guanli

数据库管理(database administration) 为保证数据库系统的正常运行和服务质量,有关人员须进行的技术管理工作,例如数据库的建立、监控和维护等。负责这些技术管理工作的个人或集体称为数据库管理员(DBA)。数据库管理的主要内容有:数据库的建立、数据库的调整、数据库的重组、数据库的重构、数据库的安全控制、数据的完整性控制和对用户提供技术支持。

数据库的建立 数据库的设计(参见数据库设计)只是提供了数据的类型、逻辑结构、联系、约束和存储结构等有关数据的描述。这些描述称为**数据模式**。要建立可运行的数据库,还需进行下列工作:①选定数据库的各种参数,例如最大的数据存储空间、缓冲块的数量、并发度等。这些参数可以由用户设置,也可以由系统按默认值设置。②定义数据库。利用**数据库管理系统(DBMS)**所提供的**数据定义语言**和命令,定义数据库名、数据模式、索引等。③准备和装入数据。定义数据库仅仅建立了数据库的框架,要建成数据库还必须装入大量的数据,这是一项浩繁的工作。在数据的准备和录入过程中,必须在技术和制度上采取措施,保证装入数据的正确性。计算机系统中原已积累的数据,要充分利用,尽可能转换成数据库的数据。

数据库的调整 数据库的设计只能反映用户初始的、主要的需求。用户的需求往往是变化的,设计的结果也难免在个别地方偏离实际。因此在数据库运行以后,应根据实际的运行情况,应用需求的变化和用户的反映对数据库做一些调整,这是 DBA 的经常任务之一。调整的内容一般有:①调整概念模式和外模式,例如将某些关系分解或合并,为某些常用的查询建立快照(即事先生成查询结果),为用户定义新的视图或修改原有的视图等。②调整索引或簇集。数据库运行以后,由于应用需求和数据特性的变化,原来设置的索引和簇集有些要撤销,有些要增补。③调整数据库运行环境,例如调整磁盘存储空间的分配,分散访问频率特别高的所谓热点数据到不同磁盘驱动器的磁盘上,将访问速度要求特别高的数据移至高速磁盘上,调整数据库参数等。

数据库的重组 数据库经过一般时间运行后,由

于存储空间分配零散,废块和废元组的增加,簇集特性变坏等原因,数据库的性能会下降,需要对数据库重组。重组的办法是先将数据库卸出到其他存储媒体或存储区,再按所定义的数据模式重新装入到数据库中。重组很费时间,且影响数据库的正常运行。因此,重组的周期和时机要合理选择,一般利用数据库比较空闲的时间进行。

数据库的重构 数据库的概念模式应是相对稳定的,但随着应用环境和需求的变化,也要求在原来设计的基础上进行扩充和修改。这个过程称数据库重构。数据库重构需在 DBA 统一规划下进行。现代的 DBMS 一般具有动态修改数据模式的功能,但重构是一个可能产生错误和有待验证的过程,边重构、边运行是不现实的。一般在运行原数据库的同时,建立新的数据库。待新数据库建立并经验证后,再将应用程序转移到新数据库上。

数据库的安全控制 保护数据库的安全是 DBA 的重要职责之一。这方面的工作有:①授予用户访问权限,定期要求用户更换口令,对密级高的数据加密,对一些敏感的数据设置审计尾迹(即记下访问这些数据的用户、终端、时间、操作等,以备日后审查之用)等。②定期为数据库留下后备副本,并保存在安全地点。对不同的数据按其重要性指定不同的恢复策略。一旦数据库系统发生故障,DBA 应首先将数据库恢复到一致状态,再对用户开放。③在技术和制度上采取必要的措施,防止计算机病毒的入侵。

数据的完整性控制 即保证数据的正确性和一致性。数据库的服务质量首先体现在所提供数据的质量。一个数据库如不能提供可信的数据,也就失去存在的价值。数据的质量表现在两个方面:①及时和准确地反映现实世界的状态。数据来自各个部门、个人、活动、事件和设备,必须有强制性的制度,才能保证数据能及时、准确地采集或录入,否则数据库会偏离现实世界的状态。②保持数据的一致性。数据都要遵守一定的语义约束,例如账面要收支平衡,一个航班订出的座位数不得超过它拥有的座位数……凡是遵守其语义约束的数据称为一致性数据。现代的 DBMS 一般都具有一定的完整性约束检查机制。DBA 应善于利用这些机制,对数据的完整性进行自动的监督和检查。但这还不够,还必须严格检查录入和更新的数据。

对用户的技术支持 DBA 应与用户保持密切的联系,了解用户的要求,帮助他们解决有关的技术

问题,例如编写介绍数据库系统的资料,指导或培训用户正确使用数据库,提供技术咨询和及时地向用户通报有关的情况等。

DBA 既是数据库系统的监护者,又是数据库系统与用户之间联系的纽带,也是拥有最高特权的数据库用户。DBA 应对数据库系统有足够的了解,也要对用户的需求相当熟悉,既要有一定的专业知识,又要有一定的管理能力。一个数据库系统能否成功地运行,DBA 有重要的作用。DBA 应充分利用 DBMS 所提供的各种手段,建好、管好和维护好数据库系统,为用户服务。从发展趋势来看,DBMS 将为 DBA 提供越来越多的手段,但这只会提高对 DBA 技术素质的要求,并未减轻其责任。

参考文献

王能斌,董逸生. 数据库设计与实现. 武汉:华中理工大学出版社,1991 (王能斌)

shujukuguanlixitong

数据库管理系统 (database management system, DBMS) 用于建立、使用和维护数据库的软件。它对数据库进行统一的管理和控制,以保证数据库的安全性和完整性。用户通过 DBMS 访问数据库中的数据,数据库管理员也通过 DBMS 进行数据库的维护工作。

按功能划分,数据库管理系统大致可分为 6 个部分:

模式翻译 提供数据定义语言 (DDL)。用它书写的数据库模式被翻译为内部表示。数据库的逻辑结构、完整性约束和物理储存结构保存在内部的数据字典中。数据库的各种数据操作(如查找、修改、插入和删除等)和数据库的维护管理都是以数据库模式为依据的。

应用程序的编译 把包含着访问数据库语句的应用程序,编译成在 DBMS 支持下可运行的目标程序。

交互式查询 提供易使用的交互式查询语言,如 SQL。DBMS 负责执行查询命令,并将查询结果显示在屏幕上。

数据的组织与存取 提供数据在外围储存设备上的物理组织与存取方法。这涉及三个方面:①提供与操作系统,特别是与文件系统的接口,包括数据文件的物理储存组织及内、外存数据交换方式等;②提供数据库的存取路径及更新维护的功能;③提供与数据库描述语言和数据库操纵语言的接口,包

括对数据字典的管理等。

事务运行管理 提供事务运行管理及运行日志,事务运行的安全性监控和数据完整性检查,事务的并发控制及系统恢复等功能。

数据库的维护 为数据库管理员提供软件支持,包括数据安全控制、完整性保障、数据库备份、数据库重组以及性能监控等维护工具。

基于关系模型的数据库管理系统已日臻完善,并已作为商品化软件广泛应用于各行各业。它在客户-服务器结构的分布式多用户环境中的应用,使数据库系统的应用进一步扩展。随着新型数据模型及数据管理的实现技术的推进,可以预期 DBMS 软件的性能还将更新和完善,应用领域也将进一步地推广。

参考文献

1. Cardenas A F. Data Base Management Systems. second edition. Boston: Allyn Bacon Inc., 1985
2. Date C J. Intrduction to Data Base Systems. Volume I, II. fifth edition. Addison - Wesley Publishing Co., 1992 (许卓群)

shujuki liantongxing biao zhun

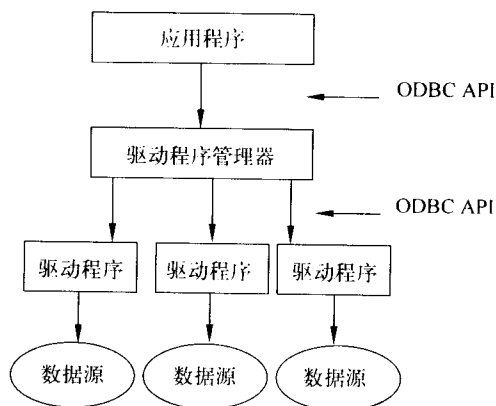
数据库连通性标准 (database connectivity standard)

不同的关系数据库管理系统 (RDBMS) 之间互访的接口标准。20 世纪 80 年代,结构查询语言 (SQL) 成为数据库语言标准,但是在 SQL 标准中,没有提供一个公共的、与数据库无关的应用程序设计接口。与此同时,RDBMS 产品迅速发展。各个数据库厂家纷纷推出各自的 SQL 软件,在遵循标准的同时又都对标准 SQL 进行了不同程度的扩充或解释,使得不同的 RDBMS 提供的 SQL 互不完全兼容。另外,不同厂商的 RDBMS 的客户系统与数据库服务器之间使用了不同的内部通信协议。因此,在某个 RDBMS 下开发的应用系统就难以访问在另一个 RDBMS 管理下的数据。

开放数据库互连 (ODBC) 是微软公司于 1991 年推出的为不同的 RDBMS 之间互访的接口,遵循 1989 年 SAC 制定的数据库调用级接口 (DBCLI) 草案。ODBC 的发展得到了工业界和第三方软件商的支持,现在已经成为一种事实上的标准 (并不是一种国际标准)。

开放数据库互连 (ODBC) 的结构包括四个主要部分: 应用程序、驱动程序管理器、数据库驱动程序和数据源,如图 1 所示。应用程序使用标准的数据

库连通性标准—应用程序接口 (ODBC API) 实现数据库的操作功能。驱动程序管理器为应用程序装载数据库驱动程序。当应用系统发出调用与数据源进行连接时,数据库驱动程序处理 ODBC 函数调用、向指定数据源发出 SQL 请求并将结果回送给应用系统。通过这四个部分使得应用程序和 DBMS 在逻辑上可以分离,应用程序与数据库无关,这样一个应用程序可以通过一组通用的代码访问不同的数据库管理系统。



ODBC API——数据库连通性标准—应用程序接口

图 1 数据库连通性标准的结构

Java 数据库互连 (JDBC) 是 **Java 语言** 的数据库访问应用程序接口,是第一个标准的、支持 Java 的数据库应用程序接口。它使得 Java 程序与数据库连接更为容易。JDBC 在功能上与 ODBC 相似,提供一个统一的数据库访问接口,因此可以用纯 Java 语言编写完整的数据库应用程序。通过使用 JDBC,可以方便地实现对几乎任何一种数据库的访问。Java 和 JDBC 的结合使得在开发数据库应用时真正实现“一旦编写,到处运行”。

参考文献

1. Geiger Kyle. Inside ODBC. Microsoft Press, 1995
2. Melton J. Understanding SQL and Java Together: A Guide to SGLJ, JDBC, and Related Technologies. Morgan Kaufmann, 2000 (顾宁 张绍华)

shujuku sheji

数据库设计 (database design) 根据用户的需求,设计数据库的结构和建立数据库的过程。

一般,数据库的设计过程可分为 5 个步骤:

(1) 需求分析 调查和分析用户的业务活动和数据的使用情况,弄清所用数据的种类、范围、数量以及它们在业务活动中交流的情况,确定用户对数据库系统的使用要求和各种约束条件等,形成用户需求规约。

(2) 概念设计 对用户要求描述的现实世界(可能是一个工厂、一个商场或者一个学校等),通过对其中信息的分类、聚集和概括,建立抽象的概念数据模型。这个概念模型应反映现实世界各部门的信息结构、信息流动情况、信息间的互相制约关系,以及各部门对信息储存、查询和加工的要求等。所建立的模型应避开数据库在计算机上的具体实现细节,用一种抽象的形式表示出来。以扩充的实体—联系模型方法为例,第一步先明确现实世界各部门所含的各种实体及其属性、实体间的联系,以及对信息的制约条件等,从而给出各部门内所用信息的局部描述(在数据库中称为用户的局部视图)。第二步再将前面得到的多个用户的局部视图集成为一个全局视图,即用户要描述的现实世界的概念数据模型。

(3) 逻辑设计 主要工作是将现实世界的概念数据模型设计成数据库的一种逻辑模式,即适应于某种特定数据库管理系统所支持的逻辑数据模式。与此同时,可能还需为各种数据处理应用领域产生相应的逻辑子模式。这一步设计的结果就是所谓“逻辑数据库”。

(4) 物理设计 根据特定数据库管理系统所提供的多种存储结构和存取方法等依赖于具体计算机结构的各项物理设计措施,对具体的应用任务选定最合适的物理存储结构(包括文件类型、索引结构和数据的存放次序与位逻辑等)、存取方法和存取路径等。这一步设计的结果就是所谓“物理数据库”。

(5) 验证设计 在上述设计的基础上,收集数据并具体建立一个数据库,运行一些典型的应用任务来验证数据库设计的正确性和合理性。一般,一个大型数据库的设计过程往往需要经过多次循环反复。当设计的某步发现问题时,可能就需要返回到前面去进行修改。因此,在做上述数据库设计时就应考虑到今后修改设计的可能性和方便性。

至今,数据库设计的很多工作仍需要人工来做,除了关系型数据库已有一套较完整的数据范式理论可用来部分地指导数据库设计之外,尚缺乏一套完善的数据库设计理论、方法和工具,以实现数据库设

计的自动化或交互式的半自动化设计。所以数据库设计今后的研究发展方向是研究数据库设计理论,寻求能够更有效地表达语义关系的数据模型,为各阶段的设计提供自动或半自动的设计工具和集成化的开发环境,使数据库的设计更加工程化、更加规范化和更加方便易行,使得在数据库的设计中充分体现软件工程的先进思想和方法。

参考文献

1. 中国大百科全书. 电子学与计算机卷. 北京: 中国大百科全书出版社, 1986
2. 萨师煊, 王珊. 数据库系统概论(第二版). 北京: 高等教育出版社, 1991 (何新贵)

shujuku xitong

数据库系统(database system) 储存、管理、处理和维护数据的软件系统。它由数据库、数据库管理员和有关软件组成。这些软件包括**数据库管理系统(DBMS)**、宿主语言、开发工具和应用程序。DBMS用于建立、使用、维护数据库。**宿主语言**是可以嵌入数据库语言的程序设计语言。数据库是长期储存在计算机中有组织的、大量的、可共享的数据集合。数据库管理员负责创建、监控和维护数据库。

数据库系统的学科含义是指研究、开发、建立、维护和应用数据库系统所涉及的理论、方法和技术。数据库系统是软件领域的一个重要分支,涉及计算机应用、软件和理论三个方面。

数据库系统的发展主要以数据模型和DBMS的发展为标志。它诞生于20世纪60年代中期。第一代数据库系统是指层次和网状数据库系统。其代表是1969年美国IBM公司研制的层次数据库系统IMS和美国数据库系统语言协会CODASYL的数据库任务组于60年代末至70年代初提出的DBTG报告。DBTG报告基于网状模型提出了数据库系统的许多概念、方法和技术。第二代数据库系统是指关系数据库系统。1970年IBM公司San Jose研究所的E. F. Codd发表了题为“大型共享数据库的关系模型”的论文,开创了关系数据库系统方法和理论的研究。目前正在研究的新一代数据库系统是数据库技术与面向对象、人工智能、并行计算、网络等技术结合的产物。其代表是面向对象数据库系统和演绎数据库系统。

数据库中的数据有逻辑和物理两方面。逻辑方面由全局模式(模式)和子模式(外模式)描述;物

理方面由存储模式(内模式)描述。这三个模式是对数据库的三级抽象,即数据库的三级结构。三级结构的采用使数据库具有较高的物理独立性和一定的逻辑独立性。数据库中的数据包括:数据本身、数据描述、数据约束和数据的存取路径。数据库独立于具体的应用,为多个应用共享,从而大大减小了数据的冗余度和不一致性。

数据库由 DBMS 统一管理,数据库的建立、使用和维护均要通过 DBMS 进行。建立数据库时,DBMS 要将用数据定义语言(DDL)写的数据库模式翻译成内部表示,数据库的逻辑结构、完整性约束和物理结构保存在数据字典中;使用数据库是通过 DBMS 提供的数据库操纵语言(DML)或查询语言进行(程序方式或交互方式);DBMS 提供的维护数据库功能包括数据库的安全性控制、完整性保障和数据库重组和数据库恢复。

数据库管理员是数据库管理机构的一组人员,具有最高的数据库用户特权,负责全面管理数据库系统,具体职责为:①建立数据库的结构;②定义数据库的安全性要求和完整性约束条件;③选择数据库的存储结构和存取路径;④监督和控制数据库的使用和运行;⑤改进数据库系统和重组数据库。

未来的数据库系统特征是处理复杂对象,开放性,智能化,方便使用。层次、网状和关系数据库系统只提供有限的、简单的数据类型,难以处理工程数据、地理数据、多媒体数据等复杂对象,所以要深入研究复杂对象的处理;系统开放要求 DBMS 有能力访问各种类型的数据库,包括 DBMS 之间的可互操作性、可连接性、易移植性等;智能化是将人工智能的方法和技术引入数据库系统,研制具有推理能力的数据库系统;方便使用是提供友善的用户界面(如 GUI 界面)。

参考文献

1. Ullman J D. Principle of Database and Knowledge-base Systems. Vol. I, II. Computer Science Press, 1989

2. 施伯乐等. 数据库系统导论. 北京: 高等教育出版社, 1994
(施伯乐 朱扬勇)

shujukuxitong sanji jiegou

数据库系统三级结构(three-level architecture of database system) 数据库系统的一种结构框架,它由外部层(单个用户的视图)、概念层(全体用户的公共视图)、内部层(存储视图)组成

(见图 1)。该结构由美国 ANSI /X3 /SPARC 的 DBMS 研究组制订。

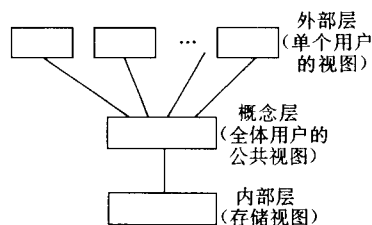


图 1 数据库系统三级结构示意图

外部层是最接近用户的层次。它是数据库的“外部视图”,是各个用户所看到的数据库。它所表示的是数据库的局部逻辑,是面向单个用户的。

内部层是最接近物理存储的层次。它是数据库的“内部视图”或“存储视图”。它与数据库的实际存储密切相关,可以理解为机器“看到”的数据库。

概念层是介于上述两者之间的层次。它是数据库的“概念视图”,是数据库中所有信息的抽象表示。它既抽象于物理存储的数据,也区别于各个用户所看到的局部数据库。概念视图可以理解为数据库管理员所看到的数据库。

数据库系统结构的外部层、概念层和内部层分别对应于数据库模式的外模式、模式和内模式。

数据库系统结构分级对于提高数据独立性具有重要意义。在三级结构间存在着两级映射。概念层-内部层映射定义了概念视图与存储的数据库之间的对应。如果存储的数据库的结构发生变化,可以相应地改变概念层-内部层映射,而使概念视图保持不变,即将存储的数据库的变化隔离在概念层之下,不反映在用户面前,因此应用程序可以保持不变,这称作**数据的物理独立性**。外部层-概念层映射定义了单个用户的外部视图与全局的概念视图之间的对应。如果概念视图发生变化,可以改变外部层-概念层映射,而使用户看到的外部视图保持不变,因此应用程序可以保持不变,这称作**数据的逻辑独立性**。

参考文献

1. Date C J. An Introduction to Database Systems. 4th Ed. Addison-Wesley, 1986

2. Ullman J D. Principles of Database and Knowledge-Base Systems. Vol. 1. Computer Science Press, 1988

(唐世渭 杨冬青)

shujuku xingneng pingjia

数据库性能评价 (database performance evaluation)

对数据库系统各项性能指标的综合评价。一般,数据库系统的性能包括其操作或应用程序的执行速度、所占存储空间、异常处理的能力、安全保密性能以及它对其所处软件环境的兼容性能等等。在评价一个数据库时,先要建立一个合适的综合评价体系,即确定各种评价因素及其相应的权重,并设计合理的综合评价方法;然后设计各种评测程序或事务元,对储存有不同数据的数据库多次执行这些评测程序或事务元,得到各评价因素的评价值;最后用综合评价方法求得评价结果。有时把数据库性能评价狭义地理解为对其执行速度的评测,这时一般用一组基准评测程序的执行时间或单位时间内能执行多少个基准评测事务元来衡量一个数据库的性能。例如,美国标准化技术研究所(NIST)为数据库语言SQL的测试开发了“SQL测试集”;事务处理性能委员会TPC开发了基准测试事务元集TPC-A、TPC-B和TPC-C。(何新贵)

shuju leixing

数据类型 (data type) 数据对象的取值集合以及对之可施行的运算集合。数据类型规定了具有该类型的变量或表达式的取值范围,也规定了与之相联系的运算的集合。

几乎每种程序设计语言都具有简单类型、构造类型和指针类型。某些程序设计语言还具有递归类型,它是借助其自身定义的类型,即在其类型定义的右方包含该类型本身的类型。递归类型的值集由借助相应类型定义方式(右方该类型的初值为空集)依次递推而得到的各个值组成。

在静态类型的语言中,所有变量和参数的类型都由程序员确定,这样每个表达式的类型都可以在编译时刻确定,所有操作都可以在编译时刻进行类型检查,大部分高级语言是静态类型语言。在动态类型的语言中,只有值的类型是确定的,变量和参数没有指定的类型,它们在运行中的不同时刻可取不同的值,这表明,在运行时执行一个运算之前必须对运算分量进行类型检查。LISP和Smalltalk是动态类型语言的例子。

在进行类型检查时,必须判别两个类型T与T'是否等价,有两种判别类型等价的方法,即按名等价与按结构等价,它们的定义是:

按名等价: $T \equiv T'$ 当且仅当T和T'在同一处中定义

按结构等价: $T \equiv T'$ 当且仅当T和T'具有相同的值集

考虑下列说明:

```

TYPE  T1 = Record      T11: Integer;
                                T12: Character

                                End;
      T2 = Record      T21: Integer;
                                T22: Character

                                End;
var    f1: T1;
        f2: T2;

procedure P(var f: T1);

```

过程调用P(f2)在按名等价方式下不能通过类型检查,因为f2的类型T2与P的参数类型T1在不同的说明中定义,但在按结构等价方式下可以通过类型检查。

由此导致类型定义两种认识,在按结构等价的语言中,类型定义仅用于建立标识符与已有类型间的绑定,而在按名等价的语言中,类型定义实质上创建了一个新的类型。

参考文献

Watt D A. Programming Language Concepts and Paradigms. Prentice Hall, 1990

(金凌紫 陈涵生)

shuju lianlu kongzhi

数据链路控制 (data link control) 为有效进行数据通信,对传输链路上的信号发送所进行的控制和管理。为此,要在物理接口之上增加进行数据链路控制的逻辑层。

数据链路控制的作用如下:

(1) 成帧 数据以帧为单位发送,每个帧必须有起始和结束标志。

(2) 流控 发送端发送帧的速率不能超过接收端接收帧的速率,以确保接收端不会发生数据溢出。

(3) 差错控制 检测和纠正帧在传输过程中可能发生的差错。

(4) 寻址 在多个线路中,识别进行信号传输的发送方和接收方地址。

(5) 信号区分 对在同一链路上传输的控制信号和数据信号加以区分。

(6) 链路管理 为建立、维护和拆除数据链路

需要进行有效的、协调的管理。

数据链路控制技术主要有**流控技术、差错检测与控制技术、以及数据链路控制规程**。

参考文献

1. 胡道元. 计算机网络(高级). 北京: 清华大学出版社, 1999

2. Stallings W. Data and Computer Communications. Prentice Hall, 1997 (胡道元)

shuju lianlu kongzhi guicheng

数据链路控制规程 (data link control protocol) 为有效地进行**数据通信**, 对传输链路上的信号发送进行控制和管理。在数据通信中, 一般用控制码对传输链路上的信号发送进行控制和管理。根据控制码的形式, 数据链路控制规程可分为面向字节规程和面向位规程两类。

(1) 面向字节规程 面向字节规程使用整个字节来表示控制码, 以字符串形式传送数据, 这种传输方式是异步的。每一个字符通过一个起始位和停止位分开, 不需要同步机制。大部分**调制解调器**中使用的异步规程与 IBM 公司的二进制同步通信规程都是面向字节规程。

(2) 面向位规程 面向位规程使用位串来表示控制码, 以位流形式传送数据, 这种传输方式是同步的。通过同步信号确定数据流中字符的开始与结束。

在**数据传输**之前, 发送者传给接收者一个同步字符, 使接收者的时钟与之同步。这种位模式的同步字符通常是特殊编码的 8 位的位串。IBM 公司的同步数据链路控制 (SDLC) 规程是面向位的, 同步字符是位串 01111110, 其后是 8 位地址、8 位控制字段以及数据。接收端的时钟提供同步信号, 并将数据分成一个个字符, 数据之后是差错检测和结束标志。高级数据链路控制 (HDLC) 规程也是面向位的规程, SDLC 是 HDLC 的子集。

高级数据链路控制规程

高级数据链路控制 (HDLC) 规程是最重要的**数据链路控制规程**。它得到广泛使用, 而且是其他许多重要的数据链路控制规程的基础。

HDLC 定义了三种类型的站、两种链路组合以及三种数据传送运行模式。

三种类型的站是: ①主站 负责控制链路的操作, 由它发出的帧称为命令; ②次站 在主站的控制下进行操作, 由它发出的帧称为响应, 主站与链路上的每个次站都分别维持一个逻辑链路; ③组合站

组合了主站和次站的特性, 既可发送命令, 又可发送响应。

两种链路组合是: ①非平衡组合 由一个主站和一个或多个次站组成, 支持全双工和半双工传输 (参见**串行传输**); ②平衡组合 由两个组合站组成, 支持全双工和半双工传输。

三种数据传送模式是: ①正常响应模式 (NRM) 用于非平衡组合, 由主站发送命令对次站的数据传送进行初始化, 次站发送数据作为对该命令的响应; ②异步平衡模式 (ABM) 用于平衡组合, 任何一个组合站都可以开始传输而不必得到另一个组合站的允许; ③异步响应模式 (ARM) 用于非平衡组合, 次站不必得到主站的允许即可进行初始化传输, 但主站仍然负责链路的初始化、纠错和断开逻辑链路等。

正常响应模式 (NRM) 用于多点链路, 例如许多终端连到一台主机。主机通过查询每个终端来控制向其输入数据。NRM 也可用于点一点链路。三种传输模式中最常用的是异步平衡模式 (ABM), 它可以提高全双工点一点链路的效率, 因为不需要查询的开销。异步响应模式 (ARM) 用得最少, 仅用于需要次站进行初始传输的场合。

高级数据链路控制规程 (HDLC) 是**同步传输**方式。所有的传输都以帧来进行, 而所有的数据和控制信息的交换也都用同一种帧的格式。

图 1 为帧的结构。在信息域之前的标识域、地址域和控制域称为报头。在信息域之后的 FCS (帧检验序列) 和标识域称为报尾。

HDLC 操作分为三个阶段: 第一阶段由某一方进行数据链路的初始化, 以便使帧的交换能有序地进行; 第二阶段双方交换用户数据和控制信息; 第三阶段由某一方终止操作。

其他数据链路控制规程

LAP B LAP B 由 ITU-T 提出, 作为 X.25 分组交换网络接口标准中的一部分 (参见**分组交换**)。它是 HDLC 的一个子集, 只提供异步平衡模式 (ABM)。专门为点一点链路设计, 其帧格式与 HDLC 相同。

LAP D LAP D 由 ITU-T 提出, 作为 ISDN 建议的一部分。LAP D 在 D 通道上提供数据链路控制, D 通道是用户 ISDN 接口处的一个逻辑通道。

LAP D 与 HDLC 有几点关键的差别。首先, 它亦只提供 ABM。其次, LAP D 只用 7 位顺序号, 16 位 CRC。最后, LAP D 用 16 位的地址域, 其中包含

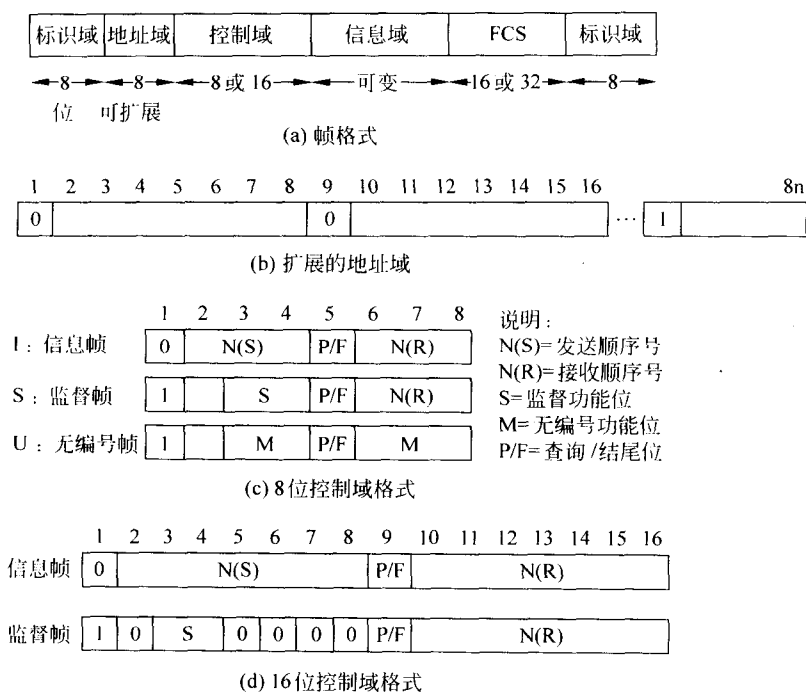


图1 HDLC 帧结构

两个子地址。一个用于标识用户方接口处的某个设备,另一个用于标识用户方接口处的某个 LAP D 逻辑用户。

逻辑链路控制(LLC) LLC 是 IEEE802 系列标准中的一部分,用于 LAN 的操作控制。它与 HDLC 既有相同之处,又有差别之处。最主要的差别在于帧的格式。LAN 的链路控制功能分为两层,底层是介质访问控制层(MAC),其上是 LLC 层。

MAC 层包含源和目的地址,在 LAN 中没有主站、次站之分,所以需用两个地址分别标识发送器和接收器。在 MAC 层用 32 位 CRC 进行差错检测,其控制域中包含一些针对介质访问控制的功能。

LLC 层中有四个域。它们是目的和源服务访问点(DSAP 和 SSAP),标识源和目的 LLC 的逻辑用户,信息域和控制域。LLC 控制域的格式与 HDLC 相同,但只用 7 位顺序号。

LLC 提供三种服务帧。连接模式服务与 HDLC 的 ABM 相同。另两种是无响应无连接模式和有响应无连接模式。

帧中继(参见帧中继交换)和异步传送模式也是常用的两种数据链路控制规程。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999
2. Stallings W. Data and Computer Communications. Prentice Hall, 1997 (胡道元)

shujuliu jisuanji

数据流计算机(dataflow computer) 以数据驱动方式进行操作的计算机。它是一种在指令操作的驱动原理上与传统的冯·诺依曼计算机根本不同的非传统计算机。

在传统的冯·诺依曼计算机中,指令执行的次序是按照指令在程序中规定的顺序进行的,因此被称为控制驱动的计算机。控制驱动计算机中的指令计数器从本质上规定了指令的串行执行,另外还规定多条指令可共用一个存储单元中的操作数,因而会产生副作用,引起指令间的数据相关,也影响指令的并行执行。数据流计算机取消了指令计数器和存储器的数据共享。在数据流计算机程序中的任一条指令,只要所需的操作数已全部齐备,就可以立即执行,一条指令的运算结果又流向下一条指令作为其操作数来启动该指令的执行,这就是**数据驱动**。这

样做可以使所有有条件执行的指令都能并行执行。如果在硬件上有足够多的执行部件、路由选择网络等资源,就能最大限度地实现指令级的并行执行,使并行处理能力高于冯·诺依曼计算机。

数据流计算机的程序适合于用有向图来表示,称为数据流图。图1是计算 $a = (b + c) + (d \times e)$ 的数据流图。图中的每一个节点表示一个操作,操作可以是一条指令,也可以是一组指令或一个功能块。节点之间通过有向弧连结。在数据流图中流动的数据,包含具体数值和该数据的流向信息,这种形式的数据称为数据标记。数据流图中还有带有布尔值的控制标记。当节点的所有输入弧上都有数据标记或控制标记时,便可执行操作,而不需指令计数器。执行后输入弧上的标记消失,运算结果出现在输出弧上,流向后继节点去启动下一个操作,这样便没有共享数据。

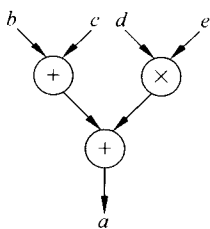


图1 数据流图举例

显然,按上述驱动原理操作,必然要执行很多不必执行的指令,例如条件转移时,两个分支的指令都要执行,这样便加重了处理部件、路由选择网络等硬件的负担,从而降低系统的效率。在后来的一些数据流计算方案中对此作了改进,使操作中只有选定数量的操作数或函数的独立变量才能用来启动操作的执行,这样可以避免不必要的计算工作。

数据流计算机有静态和动态之分。在静态数据流计算机中,每条输入弧上只允许有一个数据标记存在,不允许对一个节点作并行的多次调用,禁止对可重入代码的并行处理。它的数据流图在编译时就已完全确定,所以称为静态。这种方式实现起来简单,但限制了并行性的开拓。在动态数据流计算机中,对每个数据标记添加一个特定的标志,使每条输入弧上的数据标记可以多于一个,这样可以并行调用可重入代码,有利于并行性的开拓。它的数据流图在执行过程中是有变化的,所以称为动态。还有一种动态数据流计算机,它不加标志,由代码复制来实现可重入代码的并行调用。这种系统可以同时存

在一段可重入代码的多个副本,但要增加系统的开销。另外,还曾提出过一些不同于静态和动态数据流计算机的特殊系统结构数据流计算机。还有一些是混合驱动模式的,有的采用数据流与控制流的混合,有的采用数据流与需求流的混合。

20世纪60年代末美国麻省理工学院的Dennis及其研究组开始研究叫做VIM的VAL语言解释机,在数据流图、数据流计算语言和数据流计算机系统结构等方面做了开创性的工作。从70年代后期到80年代,数据流计算机的研究在世界上形成了热潮。研制的数字流计算机最初主要用于数值计算,后来也研制了专门用于符号处理的非数值计算的机器。这一时期有影响的项目,在静态数据流机方面有美国的Taxes分布式数据处理机、法国的LAU系统、日本的NEDIPS和IPP系统等,在动态数据流机方面有美国的MIT带标记的数据流机和DDM-1机、英国曼彻斯特数据流机、日本的SIGMA-1系统等,在非数值计算的数据流机方面有日本的EM-3机、DFM机(原始样机为EDDY)、PIM-D机和TOPSAR机等。在澳大利亚、印度等国也有研究项目。研制的机器或原始样机绝大多数是作为传统的控制驱动计算机的附加处理机,由控制驱动计算机作为主机来处理译码和输入输出任务。

在数据流计算机的研究项目中,只有极少数做出能运行的机器,大多数只做出原始样机,还有的仅仅是在传统计算机上做些模拟和分析工作。第一个能运行的机器当属DDM-1机,其他能运行的还有EDDY、曼彻斯特数据流机、LAU系统等。对于数据驱动的优点,有的还只是设想,仅经过性能分析和模拟试验,做成的一些原始样机一般也只能提供在有限条件下的实际运行分析数据。

在进入90年代后,传统计算机从性能、并行处理技术和性能价格比等方面都有长足进展,在市场上占有很大优势,数据流计算机始终无法形成产品,进入市场。现实情况促使研究人员改变纯数据流的设想,与计算机生产厂商合作研究开发混合结构的设计方案,使系统结构尽量接近传统计算机,并采用传统计算机现成的硬件和软件子系统,包括现成的微处理机。但数据流计算机的研究工作在90年代上半期仍然不得不陆续停下来。(孙强南)

shujuliu tu

数据流图(data flow diagram) 一种用于表示数据在系统运行过程中的流向与处理(加工)的

图形工具。包含五个基本成分：数据流、处理、数据存储、数据源和数据宿。“数据流”规约了系统中的数据及其流动。数据流名通常是名词或名词短语，例如“学生成绩”；“处理”表示数据的变换，用于规约系统的功能。处理名通常采用动宾结构，例如“统计学生平均成绩”；“数据存储”是数据静态结构的表示，“数据源”和“数据宿”分别是数据的产生地和归宿，它们定义了系统的边界。

数据流图作为一种系统模型的表示工具，在使用中，支持数据抽象和过程抽象；并提供分层结构，支持自顶向下逐步精化地建造系统模型。例如在结构化方法中，首先建立系统的顶层数据流图，即系统环境图，而后，通过对上一层数据流图的不断精化，得到一个可用的系统模型的表示，其中每一处理表示一个明确的功能。

(王立福)

shujuliu yuyan

数据流语言 (data flow language) 一类专门用于数据流计算机的函数式程序设计语言。其计算模型的基本原理主要有两条：①数据驱动。当且仅当指令所需的数据可用时(值全部到达，且不存在数据相关性)，该指令即执行操作；②任何指令操作均为纯函数操作。

数据流语言遵循单赋值规则，一个变量只能在赋值语句的左部出现一次。数据流语言没有函数副作用，因此，发现并行性比较容易。实际上，数据流语言和逻辑式程序设计语言是可在编译时通过数据相关性分析和全局流分析发现隐并行性的两类重要语言。

用于数据流计算机的语言可分为两类。一类是数据流机的机器语言，常称为数据流图(DFG)；另一类是数据流机的高级语言，常称为数据流语言。

DFG是一种有向图。任何一个计算任务都可用DFG描述。例如，计算 $(a/b) + (a * b)$ 的DFG，如图1所示。

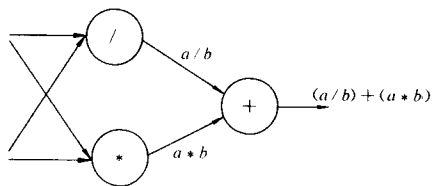


图1 计算 $(a/b) + (a * b)$

DFG直观，易于理解，刻画了数据流的一般原

理，但不便于用户使用。

数据流语言是完全基于DFG的高级语言，追求以自然方式表达最大的并行性，其主要设计目标是：①提供隐式并行，以表达问题的内在并行性；②很易转换为高度并行的DFG，在数据流机上执行；③清晰简明，模块化，有助于生成正确的程序。

数据流语言主要特点是：①是面向值的系统，抛弃了“变量”、“存储”等概念，消除了副作用。语言的活动部分(表达式、函数)具有不相干性，一旦所有输入值均为已知，则函数和表达式的执行不可能影响任何其他拟执行操作的结果；②极强的并行表达能力；③良好的“作用局部化”性质，使指令之间没有不必要的数据相关性，并可大大简化转换DFG的工作。

数据流语言的主要缺点是缺乏递归和没有通用的输入输出机制。

数据流语言的典型代表是MIT的J. B. Dennis和W. B. Ackerman设计的VAL，加州大学Arvind和K. P. Gostelow设计的ID以及法国D. Comte等设计的LAU。

参考文献

Ackerman W B. Data Flow Languages. IEEE Computer, 1982, 15(2)

(郭浩志)

shuju moxing

数据模型 (data model) 在数据库领域中定义数据及其操作的一种抽象表示。

数据模型可以由三个部分组成：①实体及实体间联系的数据结构描述；②对(表示实体和联系的)数据的操作；③完整性约束，根据适用对象的不同，数据模型可分两类。一类是面向应用的称为概念模型，这类数据模型描述用户和设计者都能理解的信息结构，强调其表达能力和易理解性，如ER模型(参见实体联系模型)；另一类模型是面向数据库管理系统的，用以刻画实体在数据库中的存储形式，称为逻辑数据模型或基本数据模型，层次模型、网状模型、关系模型等属于这一类，层次模型把实体及其间的联系表示为树形结构，网状模型把它们表示为网状结构，即有向无环图；关系模型则用平面表格的形式表示实体及其间的联系，用规则或过程等表示完整性约束，用关系间的运算作为数据操作。

参考文献

Ullman J D. Principles of Database Systems (Sec-

ond Edition). London: Pitman Publishing Ltd., 1987

(楼荣生 朱扬勇)

shuju tonglu

数据通路 (data path) 计算机中的数据、地址、指令等信息在机器内部的传送路径。用于表示计算机数据单元(包括寄存器和存储器)与操作单元(包括运算、移位、计数、传送等功能部件和门电路)之间的联系。

在设计数据通路时,不但要考虑数据传送要求,还要考虑实现方案的合理性和经济性,争取共用或借用数据路径,尽量减少直接互连线的数量。通过加法器或内部总线实现寄存器间数据的传送可节省设备,但却失去了专用路径的操作并行性。

由加法器或总线建立的公共路径,1次只能供给1对寄存器使用。如果同时有两对或两对以上的数据单元要求通过它传送数据时,需要在时间上错开。若在时间上无法错开,或为了提高操作的并行性,寄存器之间可设立专用路径。

典型的一地址指令计算机的数据通路如图1所示。围绕算术逻辑部件ALU把数据路径分为3组:

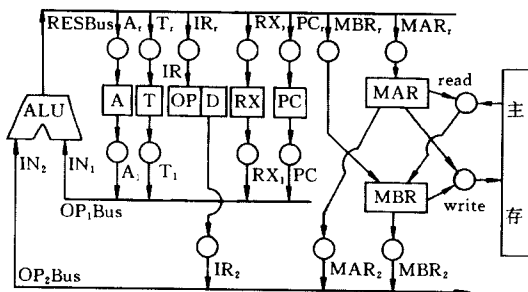


图1 典型的一地址计算机数据通路

(1) 由第一操作数总线 OP_1 Bus 供给 ALU 1 个操作数 IN_1 。它来自累加寄存器 A 或中间结果寄存器 T 或指令计数器 PC 或变址寄存器 RX, 它们分别由信号 A_1, T_1, PC_1, RX_1 控制。

(2) 由第二操作数总线 OP_2 Bus 供给 ALU 另一个操作数 IN_2 。它来自指令寄存器 IR 或存储器地址寄存器 MAR 或存储器缓冲寄存器 MBR, 它们分别由信号 IR_2, MAR_2, MBR_2 控制。

(3) 结果总线 RES Bus 可把 ALU 运算结果送给上列各寄存器, 控制信号分别为 $A_r, T_r, PC_r, RX_r, IR_r, MAR_r$ 和 MBR_r 。

ALU 输入数据的分组是按照指令要求进行划分的。ALU 完成的具体操作由操作码和时序信号配合进行控制。

例如取指令周期的操作: 指令计数器 PC 在 PC_1 控制下, 将指令地址送入 OP_1 Bus, 通过 ALU 的输出, 进入 RES Bus, 在 MAR_r 信号控制下送入存储器地址寄存器 MAR。在读命令 Read 控制下, 由主存读出 MAR 指定单元内容(指令)送入 MBR, 在 MBR_r 信号控制下将 MBR 的内容经 OP_2 Bus 送入 ALU, 经 RES Bus 在 IR_r 信号控制下将该信息送入指令寄存器 IR。

参考文献

金兰. 计算机组织与结构. 北京: 高等教育出版社, 1986

(谢树煜)

shuju tongxin

数据通信 (data communications) 两个实体间数据的传输和交换。数据通信模型如图1所示, 模型中的关键部分是:

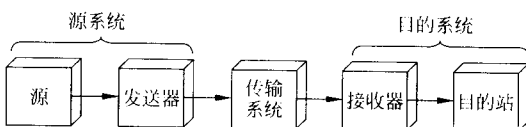
(1) 源 产生要发送的数据的设备。

(2) 发送器 对信号进行转换或编码以产生能在特定传输系统中传输的电磁信号。

(3) 传输系统 连接源和目的地的传输线或复杂的网络。

(4) 接收器 从传输系统接收信号并转换成目的站设备能处理的信号。

(5) 目的站 从接收器输入数据的设备。



(a) 数据通信模型



(b) 数据通信系统实例

图1 数据通信模型

通信系统模型描述了通信系统的基本组成和框架结构;模型也包含着许多基本概念和基础知识,并涉及大量复杂的通信技术。下面就一些最基本的概念和技术予以介绍。

(1) 传输系统利用率 有效使用传输设施。由于这些设施通常由很多的通信设备共享,因此要有

效地分配传输介质的容量,如使用多路复用技术;要协调传输服务的要求以免系统过载,如利用拥挤控制技术。

(2) 接口 为了通信,设备必须和传输系统接口,使产生的信号特性(如信号形式和信号强度)能适应传输系统的传输,以及在接收端能对数据解释。

(3) 同步 传输系统和接收设备之间、发送器和接收器之间都需要同步。接收器必须确定何时信号开始,何时信号结束,以及每个信号的间距。

(4) 交换管理 在两个实体进行通信期间的各种协调管理。

(5) 差错检测与控制 对通信过程中产生的差错进行检测和校正,并且还需要有流控的功能以防接收器来不及接收信号。

(6) 寻址和路由 决定信号到达的目的地经由的路程。

(7) 恢复 因某种原因系统被中断,而需要对系统进行恢复。

(8) 协调报文格式 两个对话实体进行协商,使报文格式一致。

(9) 安全 正确地、完整地、不被泄漏地将数据从发送器传输至接收器。

(10) 网络管理 对复杂的通信系统进行配置、监控、故障处理等管理。

数据传输是对信号的传播和处理,即将源站的数据编码成信号,经传输介质传播至目的站。数据传输的品质取决于被传输信号的品质和传输介质的特性。传输介质有双绞线电缆、同轴电缆、光缆以及无线电波等。

数据传输有串行传输和并行传输,异步传输和同步传输。发送方传到接收方的数据在同一时间内只有一位的传输方式称串行传输,同时可传输多位数据的传输方式称为并行传输。前者经济、实用,后者速度高。不用时钟等定时机制来同步发送方与接收方之间的事件的传输方式称异步传输,发送方和接收方用共同时钟来控制彼此之间的同步的传输方式为同步传输。它又可分成面向字符和面向位的两种方案,后者更为有效和灵活,已被广泛采用。

模拟信号和数字信号在通过介质传输时需进行调制和编码。数据调制是一种把模拟数据或数字数据变成模拟信号的转换技术,“调制”就是载波信号的某些特性根据输入信号的变化而变化的过程。所有调制技术都涉及到载波信号的一个或几个参数的变化:幅度、频率和相位。数据编码是一种把模拟

数据或数字数据变成数字信号的转换技术,以便通过数字通信介质传输出去,而在接收端将数字信号变成原来的形式的变换被称为解码。

为了有效利用传输系统,通过同时让传输介质携带多个传输信号来高效率地使用传输介质的传输方式称多路复用。通常有三种多路复用技术:①时分多路复用 采用分时技术,把传输线路的可用时间分成时隙,并按一定规律将时隙分配给各个输入信号的多路复用。②频分多路复用 采用分频技术,把传输线路的可用带宽分割成一个个频段,并将各个频段分配给各个输入信号的多路复用。③波分多路复用 不同波长的光载波信号在同一根光纤中同时传输的多路复用,波分多路复用光纤通信系统分为单向系统和双向系统两种。波分复用技术在通信中具有很好的应用前景。

数据通信接口是数据终端设备(DTE)和数据电路设备(DCE)之间交换数据和控制信息的接口,它必须具有为建立、维护和拆除物理链路所需的机械的、电气的、功能的和规程的特性。定义 DTE 和 DCE 之间交换数据和控制信息的接口标准为数据通信接口标准,常用的标准有 RS232C、RS 449、V.24、V.35、X.21 等。

数据链路控制是为有效进行数据通信,对传输链路上的信号发送所进行的控制和管理,是在物理接口之上增加的进行数据链路控制的逻辑层(参见数据链路控制)。

对于点到点的数据通信,只需直接完成源站到目的站的数据传输,而对于网络数据通信,就需采用数据交换技术。常用的数据交换技术有电路交换、分组交换、帧中继交换以及信元交换等,每种交换技术都有相对应的数据通信网络。

随着网络技术和网络应用的高速发展,宽带网的相关技术大量涌现(参见宽带接入)。现代数据通信网可以分成传输网、交换网和接入网三大部分。宽带传输网主要以同步数字体系(SDH)(参见时分多路复用)为基础的大容量光纤网络;宽带交换网是采用 ATM 技术(参见异步传送模式)的综合业务数字网(ISDN);宽带接入网主要有光纤接入、铜线接入、混合光纤-铜轴电缆接入以及宽带固定无线接入等(参见宽带接入技术)。

自 20 世纪 80 年代以来的计算机数据处理和数据通信的融合,大大改变了这些领域的技术和产品,推动了计算机-通信产业的发展。计算机-通信产业已经产生了以下明显的趋向:

(1) 数据、声音和视频通信已没有本质的区别。

(2) 单处理器计算机、多处理器计算机、局域网、城域网和广域网之间的界线也已变得难以区别。

这些趋向使得正在融合的计算机-通信产业从元件装配发展到系统集成,而集成的系统又发展成能传输和处理多种形式的数据和信息。技术和技术标准组织也正在走向一个能够集成所有通信的单一公共系统,使得全球所有的数据和信息源能够被方便地、相同的方式访问和获取。

参考文献

1. 胡道元. 计算机网络(高级). 北京: 清华大学出版社, 1999
2. Stallings W. Data and Computer Communications. Prentice Hall, 1997 (胡道元)

shuju tongxin jiekou

数据通信接口 (data communication interface)

数据终端设备 (DTE) 和数据电路设备 (DCE) 之间交换数据和控制信息的接口。数据通信接口必须具有为建立、维护和拆除物理链路所需的机械特性、电气特性、功能特性和规程特性。

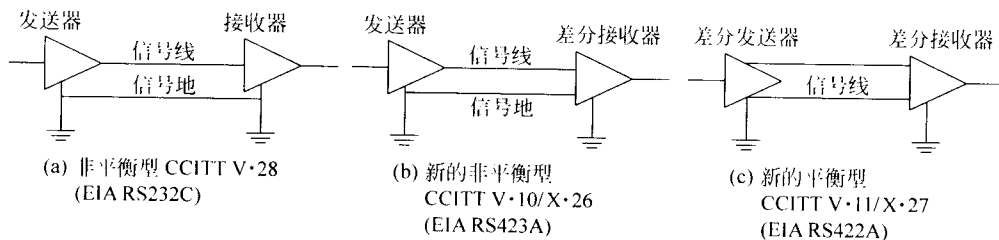


图1 物理层的电气特性

非平衡型的信号发送器和接收器均采用非平衡方式工作,每个信号用一根导线传输,所有信号公用一根地线。信号的电平是用 $+5 \sim +15\text{ V}$ 表示二进制“0”,用 $-5 \sim -15\text{ V}$ 表示二进制“1”。信号传输速率限于 20 kb/s 以内,电线长度限于 15 m 以内。由于信号线是单线,因此线间干扰大,传输过程中的外界干扰也很大。

在新的非平衡型标准中,发送器采用非平衡方式工作,而接收器采用平衡方式工作(即差分接收器)。每个信号用一根导线传输。所有信号公用两根地线,即每个方向一根地线。信号的电平是用 $+4 \sim +6\text{ V}$ 表示二进制“0”,用 $-4 \sim -6\text{ V}$ 表示二进制“1”。当传输距离达到 1 000 m 时,信号传输速

DTE 的数据传输能力有限,很少直接连接到传输线路或网络设施上,而是通过 DCE 作为中介设施连接。DCE 负责在传输介质上发送和接收信号。在传输介质上交换信号的两个 DCE 之间,以及 DTE 和 DCE 之间必须遵守相同的规程。规程包括以下四个方面:

(1) 机械特性

物理层的机械特性规定了物理连接时所采用的可插连接器的规格、尺寸,连接器中引脚的数量和排列情况等。

(2) 电气特性

物理层的电气特性规定了在物理连接上传输二进制位流时线路上信号电压高低、阻抗匹配情况、传输速率和距离的限制等。早期的电气特性标准定义物理连接边界点上的电气特性,而较新的电气特性标准定义的都是发送器和接收器的电气特性,同时还给出了互连电缆的有关规定。比较起来,较新的标准更利于发送和接收线路的集成化工作。物理层接口的电气特性主要分为三类:非平衡型,新的非平衡型和新的平衡型。图1分别描述了这三种型式的电器特性。

率在 3 kb/s 以下,随着传输速率的提高,传输距离将缩短。在 10 m 以内的近距离情况下,传输速率可达 300 kb/s 。由于接收器采用差分方式接收,且每个方向独立使用信号地,因此减少了线间干扰和外界干扰。

新的平衡型标准规定,发送器和接收器均以差分方式工作,每个信号用两根导线传输,整个接口无需公用信号地线就可以正常工作,信号的电平由两根导线上信号的差值表示。相对于其中某一根导线来说,差值在 $+4 \sim +6\text{ V}$ 表示二进制“0”,差值在 $-4 \sim -6\text{ V}$ 表示二进制“1”。当传输距离达到 1 000 m 时,信号传输速率在 100 kb/s 以下;当在 10 m 以内的近距离传输时,速率可达 10 Mb/s 。由于每

个信号均用双线传输,因此线间干扰和外界干扰大大削弱,具有较高的抗共模干扰能力。

(3) 功能特性

物理层的功能特性规定了物理接口上各条信号线的功能分配和确切定义。物理接口上的信号线一般可以分为几类:数据、控制、定时和地线。

(4) 规程特性

物理层的规程特性定义了利用信号线进行二进制位流传输的一组操作过程,即各信号线的工作规则和先后顺序。

参考文献

1. 胡道元. 计算机网络(中级). 北京:清华大学出版社,1999

2. Stallings W. Data and Computer Communications. Prentice Hall, 1997 (胡道元)

shuju tongxin jiekou biao zhun

数据通信接口标准 (data communication interface standard) 为数据终端设备(DTE)和数据电路设备(DCE)之间交换数据和控制信息而制定的规程和接口标准。

RS232C 接口

RS232C 接口是数据通信中最重要的,而且是完全遵循数据通信标准的一种接口。它的作用就是定义 DTE 设备(终端、计算机、文字处理机和多路复用机等)和 DCE 设备(将数字信号转换成模拟信号的调制解调器等)之间的接口。

RS232C 标准是目前用来连接 DTE 与 DCE 设备最流行的标准接口。大多数主要的调制解调器和 DTE 制造商都已把其设备设计得符合 RS232C 标准规格。由于电子工业协会(EIA)促进了其标准化工作,因此,RS232C 常常简称为 **EIA 接口**。

RS232C 接口完成下列两项工作:提供 DTE 与 DCE 之间的物理连接;规定它的 25 个插脚的每个信号的含义。RS232C 有四种特性:电气特性,机械特性,功能特性和规程特性。

(1) RS232C 的电气特性

接口的电气特性用来确定该接口的电压电平和电压变化的定时关系。

RS232C 标准给出以下定义:

低于 -3 V 的电压电平 = 二进制 1 = 传号

高于 +3 V 的电压电平 = 二进制 0 = 空号

DTE 和 DCE 必须使用同一个电压电平。这些电气特性确定了利用 RS232C 接口所能实现的距离

和数据速率。信号的传输距离在 RS232C 中并没有明确标明。然而,所规定的是连接 DTE 设备与 DCE 设备的电缆允许的最大容量。采用双绞线电缆连接时,距离极限约为 15 m。

(2) RS232C 的机械特性

RS232C 接口的机械特性与 DTE、DCE 实际的物理连接有关。RS232C 是一个有 25 个插脚的连接器,引线都作了具体安排。这些引线在每一端都被捆扎成一根带有端接插头的电缆。DTE 和 DCE 必须各具有一个阴阳属性相反的插头,以便与该电缆相连接。

(3) RS232C 的功能特性与规程特性

DTE 和 DCE 之间的连接必须具有数据传输功能、信号传输功能和控制功能以及其他重要的功能,这些就是 RS232C 的功能。

当 DTE 和 DCE 接收或发送数据时,需要有一条进行这种交换的通路,这种通路在 DTE 和 DCE 中被称作“引线”,当两条引线经过 RS232C 电缆连接在一起时,整个链路被称作“电路”。

图 1 示出了 RS232C 上的四个电路及其相应功能的例子。每个电路都有一个与其功能相一致的简称。

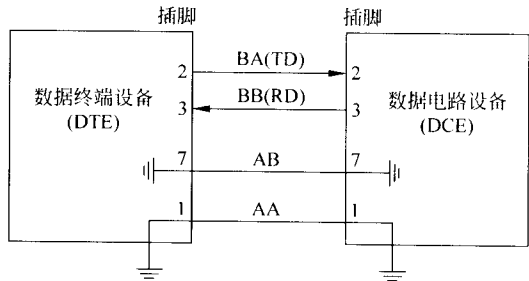


图 1 RS232C 的电路功能

- 电路 AA 保护接地线,以防设备遭到意想不到的电涌袭击(简称:机壳接地)。
- 电路 AB 信号地线,用于作为基准(参考)地(简称:信号地线)。
- 电路 BA 发送数据引线,将数据从 DTE 发送到 DCE 的引线(简称:TD)。
- 电路 BB 接收数据引线,DCE 接收数据的引线(简称:RD)。

由于 RS232C 得到了广泛使用,因此每个电路的信号说明在任何一台设备中都是很重要的。

当两个 DTE 位于 RS232C 规定的 15 m 工作极限范围之内,而且 DTE 之间未接有 DCE 时,可采用零调制解调器电缆。该电缆提供适当的阴阳匹配变

化及插脚互换位置的连接器。零调制解调器也提供一种产生通常由调制解调器产生的控制信号的手

段。图2对典型DTE与DCE的连接和使用RS232C的DTE与DTE之间的连接进行了对比。

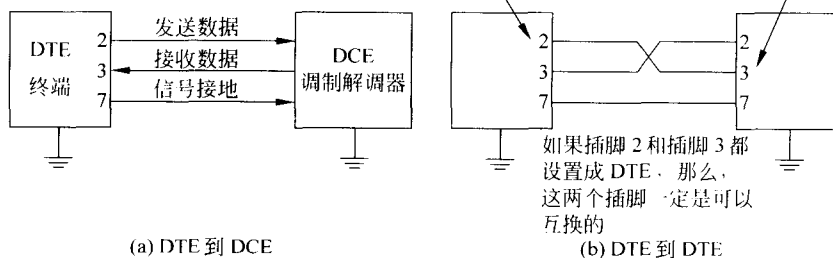


图2 零调制解调器连接

其他标准接口

V.24 标准 V.24 标准接口的功能大部分与RS232C相同。V.24描述了每个信号的操作参数,并说明了各种逻辑关系。V.24与RS232C唯一的区别在于数据终端准备引线的插脚号不同。数据终端准备引线RS232C上为插脚20,而在V.24上为电路108/1和108/2,但这个问题可以通过一个特殊的连接器来解决。

RS232C标准在北美和日本使用,V.24标准在欧洲使用。

RS449 接口 这些年来,RS232C一直还能满足数据环境对速度和距离提出的要求。但是,用户越来越希望在机构内部以更高的传输速率把数据传送到距离更远的地方,因此,要求研究出一个能满足这一功能的接口。

1975年,EIA发布了一个叫做**RS449**的规格。RS449比RS232有更多的引线,因而功能也更多。RS449标准的主要目的是将RS232C的电气特性、机械特性和功能特性分解成独立的文件。

RS449标准规格包括机械特性和功能特性,另外还有两个关于电气特性的子集标准文件:RS442(平衡式电气接口)和RS423(非平衡式电气接口)。

V.35 接口 V.35标准接口是一种CCITT标准(请注意,V.和X.标准均为CCITT推荐),应用于美国和欧洲。

V.35接口的一些特性如下:

(1) 34插脚连接器可提供附加功能,以及测试和控制电路;

(2) 与V.24的设计相类似,但V.35具有平衡的数据和时钟信号;

(3) 可用电子方法支持高比特率(直到56 kb/s),但传输距离极限则与RS232C相类似。

X.21 接口 X.21在解决接口问题时独辟蹊径。它不是对每一个功能都使用不同的引线,而是对每根引线进行多路复用。这样,每一电路都可提供很多功能。这种15插脚连接器目前总共只提供了8个电路。

下面是X.21的一些特性:

(1) 使用15插脚连接器 因具备多路复用能力而使用了较少的引线;

(2) 所连接的设备要具有适当的智能;

(3) 应用于**公用数据网** 该接口是使用另一种叫做X.25的软件接口的公用数据网的物理链路。

X.21与RS232C的本质区别在于X.21接口要求DTE和DCE具有更多的逻辑功能和智能。

如表1所示,同RS232C和RS449一样,X.21在T和R两个方向上均有发送的电路。但是,在X.21接口中,这些T和R电路不仅能提供用户数据,而且还能提供控制信息。X.21接口用于公用数据网,是较高层次的X.25接口的一个“子集”标准。

表1 X.21 电路对照表

名 称	方 向	功 能
信号接地(G)	NA	
DTE公共回线(GO)	DCE	
发送(T)	DCE	用于传送用户数据和网络控制信息,视C和I的状态而定
接收(R)	DTE	同T一样,但方向相反
控制(C)	DCE	向DCE提供控制信息(如通断信号)
指示(I)	DTE	向DTE提供指示符(如透明的数据传输阶段的开始)
信号码元定时(S)	DTE	提供位定时
字节定时(B)	DTE	提供字节(8位)定时

参考文献

1. 胡道元. 计算机网络(中级). 北京: 清华大学出版社, 1999

2. Stallings W. Data and Computer Communications. Prentice Hall, 1997 (胡道元)

shuju wajue

数据挖掘 (data mining) 从大量数据中寻找隐含在其中的规律的过程、方法和技术。又称数据库中的知识发现(KDD)。原始数据可以是结构化的,也可以是半结构化的,甚至是分布在网络上的各类数据。发现知识的方法可以是数学的,也可以是非数学的;可以是演绎的,也可以是归纳的。因此,数据挖掘涉及多门学科,它会聚了数据库、人工智能、数理统计、形象化、并行计算等领域的技术。

数据挖掘主要有数据准备、规律寻找和规律表示三个步骤。数据准备是从各种数据源中选取和集成用于数据挖掘的数据,因为这些数据一般是不完全的、有噪声的,需要清洗和整理;规律寻找是用某种方法将隐含在数据中的潜在有用的信息和知识找出来;规律表示是用尽可能符合用户习惯(如形象化)的方式将找出的规律表示出来。

1989年8月举行的第11届国际联合人工智能学术会议上首次出现数据挖掘一词。1995年在加拿大蒙特利尔市召开了第一届KDD(知识发现)国际学术会议。1997年亚太地区在新加坡组织了第一次泛太平洋KDD国际学术会议PAKDD。随后在数据库、人工智能、信息处理及知识工程等领域的国际学术刊物也纷纷开辟了KDD专题或专刊。

数据挖掘的主要技术有:

(1) 关联分析 寻找数据项之间感兴趣的关联关系。例如:我们通过对交易数据的分析可能得出“86%买‘啤酒’的人也买‘尿布’”这样一条“啤酒”和“尿布”之间的关联规则。

(2) 分类分析 找出描述并区分数据类的模型(可以是显式或隐式),以便能使用模型预测给定数据所属的数据类。例如:信用卡公司可以将持卡人的信誉度分类为:良好、普通和较差三类。分类分析通过对这些数据类的分析给出一个信誉等级的显式模型:“信誉良好的持卡人是年收入在30 000元到50 000元之间,年龄在30至45岁之间,居住面积达200m²以上的人”。这样对于一个新的持卡人,可以根据他的特征预测其信誉度。

(3) 聚类分析 把一组数据对象按照相似性归

类,即“物以类聚”,根据最大化类内的相似性、最小化类间的相似性的原则将数据对象聚类或分组。

(4) 演变分析 描述随时间变化的数据的规律或趋势,并对其建模。包括时间序列分析、序列模式分析、周期模式匹配等。例如:通过对交易数据的演变分析,可能会得到“89%情况股票X上涨一周左右后,股票Y会上涨”这样一条序列知识。

(5) 异常分析 一个数据集中往往包含一些特别的数据,其行为和模式与一般的数据不同,这些数据称为“异常”。对“异常”数据的分析称为“异常分析”。它在欺诈甄别、网络入侵检测等领域有着广泛应用。

数据挖掘虽然出现的时间不长,但技术发展迅速,软件产品日趋成熟,解决方案迅速进入各行各业,如零售业、银行、邮电、保险、医药、运输业、行政司法、生物信息处理、采矿业等。

参考文献

1. 王珊等. 数据仓库技术与联机分析处理. 北京: 科学出版社, 1998

2. Han J, Kamber M. Data Mining—Concepts and Techniques. New York: Morgan Kaufmann, 2001

(朱扬勇)

shuju wanzhengxing

数据完整性 (data integrity) 数据库中数据的准确性、正确性与有效性。

数据库中的数据完整性是用户对数据存储与维护的一种需求。它可以指定某些属性或者字段的取值必须限制在一定范围之内,也可以指定某些数据之间必须满足一定的约束条件。例如,在一个存放零件数据的数据库里,用户可以规定对任一零件,其零件号码必须惟一,零件成本必须大于零,零件价格必须大于成本等,这些都是对于数据完整性的要求。

数据完整性根据它所要求的内容可以分成不同的种类。例如,在关系数据库中就有域完整性、实体完整性,以及参照完整性等。域完整性规定了属性的取值范围,实体完整性则要求任一元组的主键码的值不得为空并且必须在所属的关系中惟一,而参照完整性则要求当一个元组的外键码值不为空时,以该外键码值作为主键码值的元组必须在相应的关系中存在。

为了保证数据完整性,数据库管理系统必须具备以下功能:①监督数据库中事务的执行并检查有

无违反数据完整性的操作。②当出现违反数据完整性的操作时,系统采取恰当的措施保证数据完整性不会被破坏。例如,一个提供实体完整性的关系数据库管理系统会在用户插入新的元组时自动检查该元组的主键码的值是否为空并且是否惟一。当主键码为空或者不惟一时,则系统会自动拒绝执行插入操作并发出警告信息。

由于数据完整性涉及到数据库中的语义维护,因此由数据库管理系统直接保证的数据完整性较为有限。为此,一些数据库管理系统提供了某种手段,例如以规则或者触发器的形式供数据库用户或管理人员对较为复杂的数据完整性加以定义。在定义中,用户或数据库管理人员说明数据完整性所涉及的对象,应当满足的完整性条件,完整性检查的时机,以及当出现违反完整性的操作时应当采取的措施。用这种方法定义的数据完整性会由数据库管理系统在数据库的使用过程中自动实施。

在更为复杂的情况下,数据库管理系统所提供的这些功能也无法满足用户对数据完整性的要求,这时必须在应用程序里加入相应的处理才可以保证所要求的数据完整性得到满足。(周立柱)

shuju yilai

数据依赖 (data dependency) 关系数据库中数据间的一种语义约束。它是通过元组属性值的相等与否来确定元组属性的相互联系。数据依赖的内容较多,主要的有函数依赖和多值依赖。

函数依赖: 设 $R(U)$ 是定义在属性集 U 上的关系模式, r 是 $R(U)$ 的任一关系, X, Y 为 U 的子集。若 r 中的任意两个元组 s, t , 在属性 X 上取值相等, 即 $s[X] = t[X]$; 则 s, t 在属性 Y 上取值也相等, 即 $s[Y] = t[Y]$, 称属性 X 函数决定属性 Y , 或属性 Y 函数依赖于 X ; 记为 $X \rightarrow Y$ 。

函数依赖蕴含在关系模式中。若 $X \supseteq Y$, 则 $X \rightarrow Y$ 成立, 称它为平凡函数依赖; 若 $X \rightarrow Y$ 成立, 且 X 中不存在真子集 X' 使得 $X' \rightarrow Y$ 成立, 则称 $X \rightarrow Y$ 为完全函数依赖, 否则称为部分函数依赖。

例: 关系模式 $R(C, S, Z)$, C 为城市名, S 为街道名, Z 为邮政编码。显然函数依赖 $Z \rightarrow C, CS \rightarrow Z$ 成立。

多值依赖: 设 $R(U)$ 是定义在属性集 U 上的关系模式, r 是 $R(U)$ 上的任意关系, X, Y 为 U 的子集。若 r 中的任意两个元组 s, t , 在属性 X 上取值相等 $s[X] = t[X]$, 则在 r 中存在元组 ϕ, ψ , 取值为:

$$\phi[X] = s[X] \quad \phi[Y] = s[Y]$$

$$\phi[U - X - Y] = t[U - X - Y];$$

$$\psi[X] = t[X] \quad \psi[Y] = t[Y]$$

$$\psi[U - X - Y] = s[U - X - Y];$$

称属性 X 多值决定属性 Y , 或属性 Y 多值依赖于 X ; 记为 $X \twoheadrightarrow Y$ 。若 $U = X \cup Y$, 则 $X \twoheadrightarrow Y$ 成立, 称它为平凡多值依赖。

例: 关系模式 $R(C, T, S, H, R)$, 其中 C, T, S, H, R 分别为课程名、教师名、学生名、上课的时间和教室。则有多值依赖 $C \twoheadrightarrow HR$ 成立。

数据依赖在关系数据库中广泛存在, 并导致了数据冗余, 使关系操作中的修改、插入、删除产生异常现象。数据依赖是关系数据库的重要研究内容。关系数据库的设计, 特别是规范化理论是建立在它的基础上的。

参考文献

1. Ullman J D. Principle of Database Systems. London: Pitman Publishing Ltd., 1982

2. Date C J. An Introduction to Database Systems. 4th Ed. Addison - Wesley, 1986

(施伯乐 朱扬勇)

shuju zhongduan shebei

数据终端设备 (data terminal equipment, DTE) 参见数据电路设备。

shuju zhunbei shebei

数据准备设备 (data preparation device)

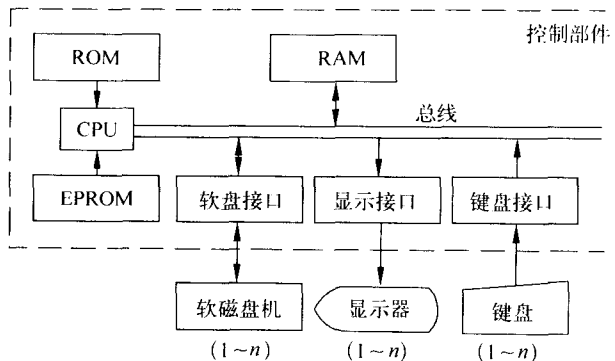
用于数据在输入计算机前进行收集、整理的脱机设备。数据输入有两种方法。一种是直接输入, 如用键盘人工键入, 用光学字符阅读机 (OCR) 或光学标记阅读机 (OMR) 等输入。另一种是间接输入, 即使用数据录入设备将数据录入到某种媒体 (例如纸带、纸卡片、磁带、软磁盘、光盘等) 上, 再从该媒体输入计算机。数据准备设备用来完成数据输入计算机前的各种准备工作。例如用卡片输入数据, 首先要用卡片穿孔机将数据穿在卡片上, 再用卡片校对机校对, 然后经卡片分类机分类, 形成一叠准备好的卡片, 最后才通过卡片输入机把数据输入到计算机中去。因此, 卡片穿孔机、卡片校对机、卡片分类机都属于数据准备设备, 它们都与主机没有直接的联系。

常见的数据录入设备是键盘、软磁盘录入设备。

由于软磁盘成本低,具有使用方便、可重复使用、易复制等优点,应用十分广泛。

键盘-软磁盘录入设备由键盘、显示器、软磁盘机和以微处理器为核心的控制部件组成(如图1所示),它允许多人同时分别操作。键盘输入的数据

显示在各自的屏幕上,便于校对和修改。数据由录入设备按一定格式编辑成文件,存储在软磁盘上,以供输入之用。



CPU——中央处理器; ROM——只读存储器;
RAM——随机存取存储器; EPROM——可擦编程只读存储器

图1 键盘-软磁盘录入设备结构原理图

利用数据录入设备使收集数据的工作可以不在一个地方进行,比如用软磁盘分别录入,然后集中到一处输入计算机,给工作提供很多方便。

过去供大型计算机数据录入的设备是专用的。由于微型计算机的普及,目前都已用微型计算机来作数据录入设备。同时,由于光盘的普及,也出现了利用光盘作媒体的数据录入设备。(林兼)

shuli luoji

数理逻辑 (mathematical logic) 用数学方法研究的逻辑。又称符号逻辑。所谓数学方法是指数学采用的一般方法,包括使用符号和公式,使用已有的数学成果和方法,特别是使用形式的公理方法。其思想可溯源到17世纪的G. W. Leibniz。他试图建立一种精确的、普遍的符号语言,并寻求一种推理演算,以使用演算来解决推理的问题。数理逻辑是由一些数理逻辑的先驱者沿着Leibniz的思想进行了实质性的工作后,而逐步完善和发展起来的。

公理方法是数学中广泛使用的一种方法。数理逻辑采用完全形式化了的公理系统,即形式系统。形式系统由它的语言、公理和推理规则三部分构成。

形式系统的语言一般采用人工语言,又称为形式语言。建立一种形式语言,首先要规定语言的符号。由这些符号根据某些确定的形式规则而得到的

符号的无穷序列,称为公式。这些形式规则就是公式的形成规则。事实上,公式是人们希望经过解释有意义的形式语言中的无穷序列。

形式系统的公理是需要系统加以肯定的论断的公式,并称这样的公式为有效公式。例如,命题演算的有效公式是重言式(参见命题逻辑),谓词演算的有效公式是逻辑有效式(参见一阶逻辑),群论的有效公式是在每个群中都解释为真命题的公式。当然,有效公式很多,只取其中一部分作为公理,为了推出其余的有效公式,需要引进推理规则。推理规则说明,如何由某些公式(称为该规则的前提)获得一个公式(称为该规则的结论)。公理和推理规则一旦给定,形式系统的全部定理就完全确定了。形式系统的定理归纳定义如下:①每个公理都是定理;②若推理规则的前提都是定理,则其结论是定理;③每个定理都可通过有穷次应用①和②获得。如果推理规则有有穷多个前提,则称该规则是有穷的。例如,分离规则是有穷的,因为它只有两个前提。命题演算和谓词演算的推理规则也都是有穷的。在大多数情况下,只使用有穷规则。但有时需要非有穷规则,例如在无穷长语言逻辑中。

如果公理集是递归集,每条推理规则对应的判定问题是可判定的,即可以能行地确定是否可用该规则由一组公式得到一个公式,则定理集是递归可枚举集(参见可计算性理论)。称定理集是递归集

的形式系统为可判定的,称定理集是递归可枚举集的形式系统为半可判定的。例如,命题演算是可判定的,谓词演算是半可判定的。

一个形式系统由它的符号集、公式的形成规则、公理集和推理规则集完全确定。形式系统本身是一个纯语法对象。形式系统的解释或意义称为语义。通常,形式系统都有逻辑推理系统或数学公理系统作为它的背景。

形式系统的协调性、独立性、可靠性和完全性是数理逻辑中经常研究的元理论性质。协调性和独立性给出了形式系统的语法性质。如果至少有一个公式不是形式系统的定理,则称该系统是协调的。如果形式系统没有多余的公理和推理规则,即只要去掉任何一个公理或推理规则,系统的定理就会减少,则称该系统是独立的。例如,命题演算和谓词演算都是协调的。可靠性和完全性给出了形式系统的语法性质和语义性质之间的关系。如果形式系统的定理都是有效的公式,则称该系统是可靠的。反之,如果有效的公式都是形式系统的定理,则称该系统是完全的。建立可靠且完全的形式系统常常是逻辑学家向往的目标,但并非总是能够如愿以偿。例如,对于高阶逻辑就不存在可靠且完全的形式系统。

用形式化方法处理逻辑推理(特别是数学中所用的推理)的思想是数理逻辑最初的思想。20世纪,由于数学奠基问题的研究而形成了四个数理逻辑分支,即模型论、公理集合论、递归论和证明论,它们构成了现代数理逻辑的主要内容。

模型论是研究形式语言和对它的解释(结构)之间的关系的理论。一个形式语言的解释称为此语言的一个结构(模型)。这是一个具有若干运算、关系及特指元素的非空集合,也称为泛代数。所以模型论又被形容为“泛代数+逻辑”。

如果在研究模型论时关心的预定模型是集合论,那就进入了公理集合论的范围。不能把公理集合论看作模型论的一个分支。各门数学中的概念、定义、定理和证明都可以用集合论语言来表达。因此,奠定数学基础就成为奠定集合论基础,归结成为公理集合论的研究。G. Cantor 于19世纪70年代创立了集合论。B. Russell 于1902年发现了罗素悖论。为了排除罗素悖论以及其他悖论,E. Zermelo 于1908年提出了集合论的第一个公理系统。后经过A. Fraenkel 的改进成为ZF系统。另一个广泛应用的集合论公理系统是J. von Neumann, P. Bernays 和K. Godel 提出的GB系统。到目前为止,在公理集合

论中还没有发现悖论。

证明论也称元数学。它是以数学证明作为研究对象的数学。20世纪20年代初,D. Hilbert 提出了一个论证数论、集合论或者数学分析的协调性的方案。他提出首先将这些理论形式化,然后按照有穷方法证明这些形式系统是协调的。1931年Godel 发表了不完全性定理。该定理指出,一个半可判定的协调的数学理论,只要包括初等数论,那么这个理论就是不完全的,即总有闭公式 A 使得 A 和 $\neg A$ 都不是该理论的定理。哥德尔第二不完全性定理还说,如果一个包含初等数论的数学理论是协调的,则其协调性不能在该理论中证明。这说明了希尔伯特方案是行不通的,使证明论的内容发生了变化。哥德尔不完全性定理揭示了形式化方法的局限性,对数学、计算机科学和其他科学的发展都产生了深远的影响。

递归论也称算法论,是研究算法、可计算性和不可计算性的数学。它研究算法的一般规律,研究数学的问题类是否存在算法解。一类问题存在算法解就是这类问题都是可解决的;一类问题不存在算法解,就是这类问题不都是可解决的。研究问题类的可解决性和不可解决性就是数理逻辑中的判定问题,这是关系到全部数理逻辑的问题。递归论与构造性数学不可分,是构造性数学的理论基础。

从最广义的角度上看,数理逻辑除包括以上内容外,还包括归纳逻辑、模态逻辑、多值逻辑、时态逻辑等,它仍然是用数学方法研究的逻辑。所以,数理逻辑包括了演绎逻辑与归纳逻辑、单调逻辑与非单调逻辑、二值逻辑与多值逻辑、一阶逻辑与高阶逻辑等在经典与非经典意义上的逻辑。演绎逻辑是研究演绎推理的逻辑,如谓词逻辑。归纳逻辑是研究归纳推理的逻辑,对于单调逻辑,增加公理或推理规则不会使推得的定理减少。对于非单调推理,增加公理或推理规则可能会使原来可以推出的定理不再能推出。多值逻辑研究多于二值的逻辑演算,即三值以至更多值的逻辑演算。对于一阶逻辑演算中的一阶谓词和一阶函词进行量化,这样就有由一阶逻辑演算扩充成为高于一阶的二阶逻辑演算了。类似地,对于任何正整数 n ,可以有 n 阶逻辑。

在进入20世纪70年代后,由于科学技术的发展,在各研究领域中都涉及了思维逻辑规律的问题。数理逻辑及其应用正迅速地发展。例如,用于数学的无穷长语言逻辑、高阶逻辑、具有广义量词的逻辑;用于哲学和社会科学的道义逻辑、存在逻辑、断

定逻辑;用于物理学的量子逻辑等。值得指出的是,数理逻辑与计算机科学有着十分密切的关系,它是反映现代数理逻辑基本特点的重要方面。

在 Leibniz 的思想中,数理逻辑、数学和计算机三者均出于一个统一的目的,即思维过程的演算化、计算化以至在计算机上实现。早在 20 世纪 30 年代,数理逻辑将推理化为一些简单机械的动作,提出了图灵机这一计算机的抽象模型,并证明了存在通用图灵机,这正是 40 年代出现的存储程序计算机(即冯·诺依曼计算机)的理论原型。

推理与计算是相通的,数理逻辑的研究成果许多都可用于计算机科学。例如,原则上数理逻辑已给出了哪些思维过程可以借计算机实现。同时,计算机科学的深入研究推动了数理逻辑的发展。例如,一阶逻辑中没有时间的概念,而关于程序的推理是涉及过程的,因此需要增加程序算子或其他包含时间概念的算子,以便适用于过程的推理。

数理逻辑倡导的形式化方法已广泛渗入到计算机科学的各个领域,如软件规范、形式语义学、程序变换、程序正确性证明、硬件综合和验证等。程序逻辑、算法逻辑、动态逻辑、时态逻辑在形式化方法中有许多应用。在人工智能科学中,人们用数学方法研究非单调推理,以使用计算机模拟人的思维,例如非单调逻辑等。相信在科学技术的数学化的发展中,数理逻辑将会不断地起着关键的作用。

参考文献

1. Kleene S C 著. 元数学导论. 莫绍揆译. 北京: 科学出版社, 1984
2. 王浩. 数理逻辑通俗讲话. 北京: 北京出版社, 1981

3. Barwise J. Handbook of Mathematical Logic. Amsterdam: North-Holland Publishing Company, 1977

(何自强 王戟)

shuli tongji

数理统计 (mathematical statistics) 以概率论为基础的一门数学分科。对所研究的随机现象进行多次观测或试验(称为取样),研究如何合理地安排试验以获取数据资料(称为试验设计),并根据所得的数据推断所研究对象的统计特性(称为统计推断或预测),直至为采取一定的决策或行动提供依据和建议的学科。

用数理统计方法解决实际问题时,一般步骤为:建立模型、收集或整理数据、统计推断、预测和决策。

所谓建立模型,是指对所研究问题的总体服从某种分布作出假定。这要依赖于对所研究现象的专业知识、历史经验以及概率论的知识。比如在研究电子元件(经过老化试验)的寿命时,一般假定它们服从指数分布,而当考虑射击误差时,即可认为落点服从高斯分布。一般情况下,则可将测量数据描绘在各种分布的概率纸上,并与某一分布曲线比较,如果接近就认为总体服从这种分布,这就是数理统计中的非参数检验。

数据的收集一般有普查、抽查和设计专门试验三种方式。普查的结果没有随机性,不属于数理统计研究的范围,现实中许多数据是不能普查的,比如产品的寿命,人们不能够为取得数据而把产品消耗光。数理统计关心的是抽查,也就是从产品中抽取一部分,观察其某种指标,从而推断全部产品的性质。关于如何进行抽样的研究构成了数理统计的一个分支,即抽样论。为了某一目的,专门安排试验去获得数据,这在数理统计中也是常见的。比如为了研究农作物的产量与肥、水、光照、土壤的关系,可以专门设计各因素在特定的各种量下的试验,正交试验法就是这样一种优化的试验设计方法。

有了数据后,再把它用适当的方式制成图表,比如样本的直方图,由此可以看出一些粗略的规律性,并计算其样本均值、样本方差,以供进一步做统计推断。

统计推断包括参数估计、假设检验、回归分析、方差分析等内容,这些是数理统计的主题,在此基础上进行统计预测,比如用回归方程来预测量的变化。统计决策是在统计推断的基础上,对人们决策的一种建议。比如一个商店应该考虑商品存储的最佳数量,太少,供不应求,对商店是一个损失;太多,造成积压,不但存储本身需要费用,而且由于存储过长造成的损坏及资金的积压都是一种损失。我们可以把带有随机性的诸多因素综合在一起,考虑商店的经济效益。统计决策可以帮助我们得出各种商品的最佳存储量,这方面的研究已经发展成为储存论,它是统计决策的一部分。下面列出数理统计中经常要遇到的最基本概念。

母体(或总体) 指研究对象的全体。比如欲研究某灯泡厂生产灯泡的使用寿命,则把该生产厂在相同生产条件下生产的灯泡寿命全体看成是一个母体,它是一个随机变量。

样本(或子样) 对母体作 n 次独立的观察,得到 n 个随机变量 X_1, \dots, X_n , 它们相互独立,且与母

体 X 同分布。称如此的 X 是一个样本容量为 n 的样本(或子样)。仍以灯泡的寿命为例,如取 $n=3$, 即任取 3 个灯泡作寿命试验,得到三个数 (x_1, x_2, x_3) , 这三个数就是样本 (X_1, X_2, X_3) 的一次观察值,而 (X_1, X_2, X_3) 本身就是三元的随机变量。

统计量 设 (X_1, \dots, X_n) 为来自母体 X 的子样, h 为 n 元变量的已知函数,如果 $T = h(X_1, \dots, X_n)$ 仍是一个随机变量,则称 T 为统计量。一般地, h 是连续或分段连续的已知函数,此时 T 必为统计量。

χ^2 分布 若 ξ_1, \dots, ξ_n 为 n 个相互独立且服从 $N(0,1)$ 分布的变量,则 $\chi^2 = \sum_{i=1}^n \xi_i^2$ 的密度函数为:

$$f_{\chi^2}(x) = \begin{cases} \frac{1}{2^{\frac{n}{2}} \Gamma(\frac{n}{2})} x^{\frac{n}{2}-1} e^{-\frac{x}{2}}, & \text{当 } x \geq 0 \\ 0, & \text{其他} \end{cases}$$

此时称随机变量 χ^2 服从自由度为 n 的 χ^2 分布,简记为 χ_n^2 。

设母体 $X \sim N(\mu, \sigma^2)$, 子样 (X_1, X_2, \dots, X_n) 来自母体 X , 子样均值和子样方差分别为

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i, \quad s^2 = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2$$

则可知 $\bar{X} \sim N(\mu, \frac{\sigma^2}{n})$, $\frac{ns^2}{\sigma^2} \sim \chi_{n-1}^2$, 且 \bar{X} 与 s^2 相互独立。

t 分布 设 ξ, Z 相互独立, $\xi \sim N(0,1)$, $Z \sim \chi_n^2$, 则 $t = \xi / \sqrt{\frac{Z}{n}}$ 服从自由度为 n 的 t 分布,简记为 t_n , 其密度函数为:

$$f_t(x) = \frac{\Gamma(\frac{n+1}{2})}{\sqrt{n\pi} \Gamma(\frac{n}{2})} \left(1 + \frac{x^2}{n}\right)^{-\frac{n+1}{2}}$$

F 分布 设 $\xi \sim \chi_m^2, \eta \sim \chi_n^2$ 且 ξ 与 η 相互独立, 则

$$\zeta = \frac{\xi/m}{\eta/n}$$

服从自由度为 m 及 n 的 F 分布,简记为 $F(m, n)$, 其密度函数

$$f_\zeta(x) = \begin{cases} 0 & \text{当 } x \leq 0 \\ \frac{\Gamma(\frac{m+n}{2})}{\Gamma(\frac{m}{2}) \Gamma(\frac{n}{2})} m^{\frac{m}{2}} n^{\frac{n}{2}} \frac{x^{\frac{m}{2}-1}}{(mx+n)^{\frac{m+n}{2}}} & \text{当 } x > 0 \end{cases}$$

上述 χ^2 分布, t 分布, F 分布在统计中应用得很多,在统计用表中给出了这些分布的临界值表,供大家使用。

参考文献

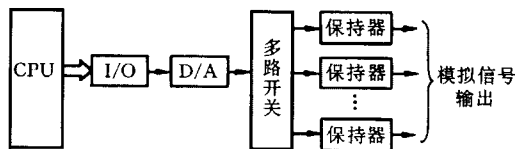
1. 王梓坤. 概率论基础及其应用. 北京: 科学出版社, 1976
2. 陈希孺. 数理统计引论. 北京: 科学出版社, 1981 (金治明)

shumo zhuanhuanqi

数模转换器 (digital - to - analog converter, DAC) 把数字信号转换成模拟信号的设备。当计算机需要控制某些物理量(如温度、压力、流量、位移等)时,首先要将经过计算机处理形成的二进制离散的数字量转换成连续的用电压或电流表示的模拟量。数模转换器就是执行这种转换的设备,它把数字量作为输入量,实时产生与数字量相对应的模拟量。

数模转换器是工业控制中重要的输出设备。在多媒体计算机中,用数模转换器电路把从磁盘或光盘中读出的数字信号转换成用模拟量表示的音频或视频信号,重现音乐或图像。**数模转换器电路**目前已有多种专用芯片。转换方式多采用 T 形电阻解码网络。数字量的输入可以是串行的,也可以是并行的。输出量有的用电压,有的用电流,视使用要求而定。和模数转换器一样,数模转换器可以看成是计算机的一条输出通道。除了单通道之外,它有两种多通道的形式:

(1) 多通道共享数模转换器电路(图 1) 它适用于不需要同时输出多路模拟信号的场合。



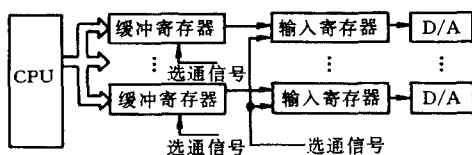
CPU —— 中央处理器; I/O —— 输入输出电路;

D/A —— 数模转换器

图 1 多通道共享 D/A 系统

(2) 同时输出多路模拟信号的通道(图 2) 采用这种结构时,计算机依次把数字量送入各路缓冲寄存器,经过数模转换器电路转换后,同时输出。

同模数转换器一样,数模转换器除了硬件之外,



CPU——中央处理器；D/A——数模转换器

图2 模拟信号可以同时输出的多通道 D/A 系统

还要有必要的软件的支持。

参考文献

1. 王秀玲等. 微型计算机 A/D, D/A 转换接口技术及数据采集系统设计. 北京: 清华大学出版社, 1984
2. 刘乐善等. 微型计算机接口技术原理及应用. 武汉: 华中理工大学出版社, 1996 (林兼)

shuzhi bijin

数值逼近 (numerical approximation) 关于如何使用容易数值计算的函数来近似地代替任意函数的方法与过程。

对于给定的较广泛的函数类 F 中的函数 f , 从较小的子类 H 中寻求一个函数 h , 使之在某种意义下近似代替函数 f , 以便于 f 的计算与处理。函数逼近的主要内容有: 对于某些特定的被逼近函数类 F 与逼近函数类 H , 讨论逼近的可能性; 最佳逼近的存在性、特征、惟一性、误差估计以及逼近算法等。它是现代数值分析的基本组成部分, 除自身具有独立学科分支的意义外, 还可用于构造数值积分、求函数零点、解微分方程和积分方程等的近似方法。

设被逼近函数 $f \in C[a, b]$, 逼近函数类记作 $H \subset C[a, b]$, 用

$$\|f - g\|_p = \left[\int_a^b |f(x) - g(x)|^p \omega(x) dx \right]^{\frac{1}{p}} \quad (1 \leq p < +\infty) \quad (1)$$

来度量函数 f 与 g 的距离, 其中 $\omega(x) > 0$ 为选取的权函数。特别, 当 $p = \infty$ 时, 通常取 $\omega(x) \equiv 1$, 此时

$$\|f - g\|_\infty = \max_{a \leq x \leq b} |f(x) - g(x)|$$

这种度量 F 的逼近称为一致逼近; 另一种重要情形是当 $p = 2$ 时, 称为均方逼近或平方逼近。

最佳逼近 若 $h \in H$ 满足

$$\|f - h\|_p = \inf_{g \in H} \|f - g\|_p$$

则称 h 为距离度量 (1) 意义下 f 在 H 中的最佳逼近。对于 $p = 2$ 和 ∞ 时, 相应的 h 分别称为 f 在 H 中的最佳平方逼近和最佳一致逼近, 后一种情况又称切比雪夫逼近, 它是由 П. Л. Чебышев 在 1854 年首先开始研究的。

多项式逼近 指 H 取作多项式类的情形。1885 年 Weierstrass 的著名定理给出了关于用多项式一致逼近连续函数到任意精度的可能性: 若 $f \in C[a, b]$, 则对任意 $\varepsilon > 0$, 都存在代数多项式 p , 使 $\|f - p\|_\infty < \varepsilon$ 。它同时给出了用三角多项式一致逼近周期连续函数到任意精度的结果。H. L. Lebesgue 曾证明了对于连续函数类而言, 切比雪夫级数展开式的部分和是最佳一致多项式逼近的很好近似。

有理逼近 指 H 取作

$$R_m^n[a, b]$$

$$= \left\{ R = \frac{P}{Q} \mid \partial P \leq n, \partial Q \leq m, Q > 0, \frac{P}{Q} \text{ 不可约} \right\}$$

的情形, 其中 P, Q 为多项式函数, $\partial P, \partial Q$ 表示它们的次数, 当 $f \in C[a, b]$ 时, f 在 $R_m^n[a, b]$ 中的最佳一致逼近 R^* 存在且惟一的充要条件是: 在 $[a, b]$ 上有一组点 $a \leq x_1 < x_2 < \cdots < x_k \leq b$ ($k \geq \max\{n + \partial Q, m + \partial P\} + 2$) 使得误差 $R^* - f$ 在这些点上达到其最大绝对值, 而且符号正负交替变化, 即

$$\begin{aligned} R^*(x_i) - f(x_i) &= (-1)^i \lambda \\ (i &= 1, 2, \cdots, k) \\ |\lambda| &= \|R^* - f\|_\infty \end{aligned}$$

一种特殊的有理逼近称为帕德逼近, 它应用于数值分析的一些领域以及物理学和化学的某些计算

问题中。设 $f(x) = \sum_{i=0}^{\infty} \alpha_i x^i$, 若存在多项式

$$P_n(x) = \sum_{i=0}^n p_i x^i$$

$$Q_m(x) = \sum_{i=0}^m q_i x^i$$

满足条件 $f(x) - P_n(x)/Q_m(x) = O(x^{n+m+1})$ 且 P_n/Q_m 不可约, $Q_m(0) = q_0 = 1$, 则称 $P_n(x)/Q_m(x)$ 为 $f(x)$ 在点 $x = 0$ 处的 (n, m) 阶帕德逼近, 并简记为 $[n/m] = P_n(x)/Q_m(x)$ 。若 $[n/m]$ 存在则必惟一。已证明帕德逼近是最佳局部有理切比雪夫逼近。帕德将 $[n/m]$ 依自然顺序排列成表格

[0/0]	[0/1]	[0/2]	[0/3]	...
[1/0]	[1/1]	[1/2]	[1/3]	...
[2/0]	[2/1]	[2/2]	[2/3]	...
[3/0]	[3/1]	[3/2]	[3/3]	...
⋮	⋮	⋮	⋮	

称为帕德表,实践表明,帕德表中主对角线上及其两侧的元素 $[m/m]$ 和 $[n/n \pm 1]$, 在 $n+m$ 相等的诸元素中,通常有更好的逼近效果。

参考文献

1. Натансон И П 著. 函数构造论. 徐家福等译. 北京: 科学出版社, 1965

2. Cheney E W 著. 逼近论导引. 徐献瑜等译. 上海: 上海科学技术出版社, 1981 (沈毅)

shuzhi jifen

数值积分 (numerical integration) 从近似计算角度出发,研究如何采用有效的数值计算过程求定积分近似值的方法与理论。通常求定积分

$$I = \int_a^b f(x) dx \quad (1)$$

的方法是先找出被积函数的原函数 $F(x)$, 然后由 $F(x)$ 求得积分(1)的值, 即 $I = F(b) - F(a)$ 。然而实际情况是:有很多的积分不能解析地求解;有些积分虽然可以解析地求解,但原函数比被积函数复杂得多,且难于给出最后的数值结果;还有不少情况,被积函数 f 没有具体表达式,无法采用解析求解。因此,在实际计算时,常采用数值积分方法。

数值积分公式 一般形如

$$\int_a^b \rho(x)f(x) dx \cong \sum_{i=0}^n A_i f(x_i) \quad (2)$$

的近似公式,为数值积分公式,也称求积公式。 $\rho(x)$ 是 $[a, b]$ 上大于等于零的权函数, x_i 和 A_i ($i = 0, 1, 2, \dots, n$) 分别称为求积结点和求积系数,式(2)右端称为求积和,差式

$$E(f) = \int_a^b \rho(x)f(x) dx - \sum_{i=0}^n A_i f(x_i) \quad (3)$$

称为求积误差,区间 $[a, b]$ 为有限,也可无限。由式(3)知,构造求积公式的问题就是确定 x_i 和 A_i , 使 $E(f)$ 在某种意义下尽可能地小。

代数精度 若式(2)对 $f(x) = x^k$ ($k = 0, 1, \dots, d$) 精确成立,即 $E(f) = 0$, 而当 $f(x) = x^{d+1}$ 时,式(2)不再是精确等式,则称式(2)的代数精度为 d 。对一般连续函数 $f(x)$ 来说, d 越大, $E(f)$ 越小,因此,可以用代数精度高低说明求积公式的优劣。

牛顿-柯特斯求积公式 考虑权函数为 1 且 $[a, b]$ 为有限时的积分,可以通过插值途径来建立数值积分公式。将 $[a, b]$ 分为 n 等分, $h = (b-a)/n$, 在 $n+1$ 个等距结点 $x_i = a + ih$ ($i = 0, 1, \dots, n$) 上,函数值为 $f(x_i) = y_i$ ($i = 0, 1, \dots, n$), 由 $f(x)$ 的 n 次拉格朗日插值多项式而建立起的积分公式为

$$\begin{aligned} & \int_a^b f(x) dx \\ &= \sum_{k=0}^n A_k f(x_k) + \frac{1}{(n+1)!} \int_a^b f^{(n+1)}(\xi) \prod_{j=0}^n (x-x_j) dx \\ &= I_n(f) + E_n(f) \end{aligned} \quad (4)$$

其中

$$A_k = \int_{x_0}^b \prod_{\substack{j=0 \\ j \neq k}}^n \frac{x-x_j}{x_k-x_j} dx, \quad k = 0, 1, \dots, n; \quad \xi \in [a, b]$$

且依赖于 $x, E_n(f)$ 为拉格朗日插值多项式余项。

若令 $x = a + th$ ($0 \leq t \leq n$), $dx = hdt$, $x_k = a + kh$, 由式(4)得到

$$A_k = C_k^{(n)} (x_n - x_0)$$

其中

$$C_k^{(n)} = \frac{(-1)^{n-k}}{nk!(n-k)!} \int_0^n \prod_{\substack{j=0 \\ j \neq k}}^n (t-j) dt$$

这时 $I_n(f)$ 和 $E_n(f)$ 分别为

$$I_n(f) = (x_n - x_0) \sum_{k=0}^n C_k^{(n)} f_k \quad (5)$$

$$\begin{aligned} E_n(f) &= \int_a^b \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{j=0}^n (x-x_j) dx \\ &= \frac{1}{(n+1)!} \int_a^b f^{(n+1)}(\xi) \prod_{j=0}^n (x-x_j) dx \end{aligned} \quad (6)$$

公式(5)称为牛顿-柯特斯公式, $C_k^{(n)}$ 称为柯特斯系数,其值依赖于 $[a, b]$ 等分数 n , 其代数精度至少为 n 。

当 $n = 1$ 时,由公式(4)得到梯形公式及其误差

$$I_1 = \frac{1}{2} (x_1 - x_0) (f_0 + f_1) \quad (7)$$

$$E_1(f) = -\frac{h^3}{12} f''(\xi), \quad \xi \in (x_0, x_1) \quad (8)$$

其中 $h = x_1 - x_0$

当 $n = 2$ 时,由公式(4)得到辛普森公式及其误差

$$I_2 = \frac{h}{3} (f_0 + 4f_1 + f_2) \quad (9)$$

$$E_2(f) = \frac{(x_2 - x_0)^5}{2880} f^{(4)}(\xi), \quad \xi \in (x_0, x_2) \quad (10)$$

其中 $h = \frac{1}{2}(x_2 - x_0)$ 。

当 $n = 4$ 时, 由公式 (4) 得到常用的柯特斯公式及其误差

$$I_4 = \frac{2h}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) \quad (11)$$

$$E_n(f) = -\frac{8}{945 \times 4^7}(x_4 - x_0)^7 f^{(6)}(\xi), \quad \xi \in (x_0, x_4) \quad (12)$$

其中 $h = (x_4 - x_0)/4$

由公式 (8), (10) 和 (12) 看出, 当积分区间很小时, 用 (7), (9) 和 (11) 三个公式计算定积分可以取得很好效果。但是当积分区间很大时, 误差较大, 这时可采用复化求积公式。

复化梯形公式 将区间 $[a, b]$ 分为 n 个小区间, 令 $h = \frac{b-a}{n}$, $x_i = a + ih$, $x_0 = a$, $x_n = b$, 在每个小区间上使用梯形公式, 并对 $i = 0, 1, \dots, n-1$ 求和, 就得到复化梯形公式及其误差

$$\int_a^b f(x) dx \approx T_n = \frac{h}{2} [f(a) + f(b) + 2 \sum_{i=1}^{n-1} f(x_i)] \quad (13)$$

$$E_1^{(n)}(f) = -\frac{b-a}{12} h^2 f''(\xi), \quad \xi \in [a, b] \quad (14)$$

复化辛普森公式 将区间 $[a, b]$ 等分为 $2n$ 个小区间, 令 $h = \frac{b-a}{2n}$, 在每两个相邻小区间上使用辛普森公式, 并对 $i = 0, 1, \dots, n-1$ 求和就得复化辛普森公式及其误差

$$\begin{aligned} \int_a^b f(x) dx &\approx S_{2n} \\ &= \frac{h}{3} [f(a) + f(b) + 4 \sum_{i=1}^n f(x_{2i-1}) \\ &\quad + 2 \sum_{i=1}^{n-1} f(x_{2i})] \end{aligned} \quad (15)$$

$$E_2^{(n)}(f) = -\frac{h^4}{180}(b-a)f^{(4)}(\xi), \quad \xi \in [a, b] \quad (16)$$

高斯型公式 考虑 $\rho(x) \neq 1$ 的情形。公式 (2) 含有 $2(n+1)$ 个参数 (x_i 和 A_i , $i = 0, 1, \dots, n$), 适当选择这些参数, 可以使公式 (2) 的代数精度达到 $2n+1$ 。数值积分理论中一个基本定理断言: 只要把结点 x_0, x_1, \dots, x_n 取为区间 $[a, b]$ 上关于权函数 $\rho(x)$ 的 $n+1$ 次正交多项式的零点, 则内插型求积公式 (2) 可达到最高代数精度 $2n+1$, 这时, 相应的求积公式称为高斯型求积公式。

设 $Q_{n+1}(x)$ 是 $[a, b]$ 上关于权 $\rho(x)$ 的 $n+1$ 次正交多项式 (即当 $i, j = 1, 2, \dots$ 且 $i \neq j$ 时成立 $\int_a^b \rho(x) Q_i(x) Q_j(x) dx = 0$, 则称 Q_i, Q_j 在 $[a, b]$ 上关于权 $\rho(x)$ 正交), 利用正交多项式的一些关系式和性质, 可以导出 x_0, x_1, \dots, x_n 是 $Q_{n+1}(x)$ 的 $n+1$ 个零点, 则高斯型求积公式的 $n+1$ 个系数为

$$\begin{aligned} A_i &= \frac{1}{Q_{n+1}^*(x_i)} \int_a^b \frac{\rho(x) Q_{n+1}^*(x)}{(x - x_i)} dx \\ &= \frac{a_{n+1}}{a_n Q_{n+1}^*(x_i) Q_n^*(x_i)}, \quad i = 0, 1, 2, \dots, n \end{aligned} \quad (17)$$

式中 $Q_{n+1}^*(x)$ 是 $[a, b]$ 上关于权函数 $\rho(x)$ 的 $n+1$ 次规格化正交多项式, a_{n+1} 和 a_n 分别是 $n+1$ 次和 n 次规格化正交多项式 $Q_{n+1}^*(x)$ 和 $Q_n^*(x)$ 的首项系数。

表 1 几种常用的高斯型求积公式

$[a, b]$	$\rho(x)$	x_0, x_1, \dots, x_n	A_i	E_n
$[-1, 1]$	1	勒让德多项式 $P_{n+1}(x)$ 的零点	$-\frac{2}{(n+2)P_{n+2}(x_i)P'_{n+1}(x)}$	$\frac{2^{n+1}[(n+1)!]^4}{(2n+3)[(2n+2)!]^3} f^{(2n+2)}(\xi)$
$[0, \infty)$	e^{-x}	拉盖尔多项式 $L_{n+1}(x)$ 的零点	$\frac{[(n+1)!]^2}{L'_{n+1}(x_i)L_{n+2}(x_i)}$	$\frac{[(n+1)!]^2}{(2n+2)!} f^{(2n+2)}(\xi)$
$[-1, 1]$	$\frac{1}{\sqrt{1-x^2}}$	切比雪夫多项式 $T_{n+1}(x)$ 的零点	$\frac{\pi}{(n+1)}$	$\frac{\pi}{2^{2n+1}(2n+2)!} f^{(2n+2)}(\xi)$
$(-\infty, \infty)$	e^{-x^2}	埃尔米特多项式 $H_{n+1}(x)$ 的零点	$-\frac{2^{n+2}\sqrt{\pi}(n+1)!}{H'_{n+1}(x_i)H_{n+2}(x_i)}$	$\frac{\sqrt{\pi}(n+1)!}{2^{2n+1}(2n+2)!} f^{(2n+2)}(\xi)$

高斯型积分公式(2)的截断误差为

$$E_n = \int_a^b \rho(x)f(x)dx - \sum_{i=0}^n A_i f(x_i) \\ = \frac{f^{(2n)}(\xi)}{a_{n+1}^2(2n)!} \int_a^b \rho(x)Q_{n+1}^2(x)dx, \quad a < \xi < b \quad (18)$$

已经证明:当 $f^{(2n)}(x)$ 在 $[a, b]$ 上连续,那么当 $n \rightarrow \infty$ 时,高斯型积分公式收敛于定积分,即

$$\lim_{n \rightarrow \infty} \sum_{i=0}^n A_i f(x_i) = \int_a^b \rho(x)f(x)dx$$

牛顿-柯特斯公式却不具备这种收敛性质。

上面对于一般权函数讨论了高斯型积分公式。不同的权函数有不同的具体高斯型积分公式。表1给出几种常用的高斯型求积公式及截断误差。用这些公式积分时,可用已给出的求积结点与系数表。

例 用勒让德求积公式计算积分 $I(f) = \int_0^\pi e^x \cos(x) dx$

$I(f)$ 的积分精确值为-12.0703463164,用勒让德求积公式计算结果为

n	$I_n(f)$	E_n
2	-12.33621046570	2.66×10^{-1}
3	-12.12742045017	5.71×10^{-2}
4	-12.07018949029	-1.57×10^{-4}
5	-12.07032853589	-1.78×10^{-5}
6	-12.07034633110	1.47×10^{-8}
7	-12.07034631753	1.14×10^{-9}
8	-12.07034631639	$< 5.0 \times 10^{-12}$

除上述方法外,还可采用龙贝格求积公式、三次样条求积公式、自适应求积公式等。奇异积分问题,一般不能用前面所述方法计算,必须针对具体积分选择适当方法。

参考文献

1. 冯康等编. 数值计算方法. 北京: 国防工业出版社, 1978
2. 关治, 陈景良. 数值计算方法. 北京: 清华大学出版社, 1990 (李晓梅)

shuzhi jisuan

数值计算 (numerical computation) 有效使用数字计算机求数学问题近似解的方法与过程, 或这些方法与过程连同有关理论所构成的学科。

数值计算是一门实用性很强的学科, 近年来随着计算机的发展和广泛应用, 许多计算领域的问题,

如计算力学、计算物理、计算化学、计算经济学等新分支都可归结为数值计算问题。

简史 早在公元前14世纪的商代就基本形成十进制的记数法, 春秋战国时代筹算已得到普遍应用, 并形成和发展了筹算作为通用有效的计算工具, 以后改进演化为算盘, 相应发展了珠算方法。

数值计算在古代已取得了重要成果。早在公元1世纪, 汉代的《九章算术》记载了开平方、开立方、解一元二次方程和三元一次方程的方法。公元5世纪南北朝时期, 著名数学家祖冲之算得圆周率 π 为3.14159265, 精确到小数点后7位, 这项纪录保持了近千年。南宋数学家秦九韶于1247年写成《数书九章》, 提出的联立一次同余式和高次方程数值解的秦九韶法, 比西方 Euler 和 Horner 于1819年提出的算法早五百多年。

随着近代数学的形成和发展, 数值计算也取得了相应的进步。20世纪40年代中后期, 电子计算机的出现与迅速发展, 有力推动了数值计算的研究与发展, 解决了许多难度与规模都很大的计算问题, 提出了许多新的数值方法, 数值计算在整个科学技术和经济生活中的重要性得到前所未有的体现, 并获得极大的发展, 形成了一门新的学科分支。

数值计算与计算机的发展相辅相成, 相互促进。大量数值计算的需要, 促使计算机体系结构及性能不断更新, 而计算机的发展又推动着数值计算方法的发展, 每当计算机发生一次变革, 数值计算也产生一次飞跃。为适应现代计算机的飞速发展, 对数值计算提出了新的要求, 对原有计算方法提出了重新评价、筛选、改造和创新。与此同时, 也涌现了许多新概念、新方法, 从而构成了现代数值计算的新含义, 如计算机系统界面的多媒体化, 给计算过程可视化提供了软件环境。

研究内容 数值计算研究有效使用计算机数值求解各种数学问题, 包括离散型方程的数值求解和连续系统离散化的数值求解, 在数值求解数学问题时, 需要考虑误差、收敛性和稳定性等问题。所谓关于给定计算问题的一个近似算法是收敛的, 是指由该算法能产生近似解的一个无穷集合, 这个集合按某种选定的距离能逼近精确解到任意程度。即对任给的 $\varepsilon > 0$, 都能从该集合中找到与精确解的距离小于 ε 的近似解。误差是指连续系统离散化产生的方法误差(截断误差)和数值计算过程中产生的误差(舍入误差)。稳定性是指在执行数值算法的过程中, 舍入误差的积累不影响产生可靠结果。此外,

还要研究算法的计算复杂性(计算量大小为时间复杂性,存储量大小为空间复杂性)以及在使用计算机时,算法的自适应性。因此,误差、收敛性、稳定性、计算复杂性和自适应性是数值计算的基本问题,刻画了数值计算方法的可靠性、准确性、效率以及使用方便性,是数值计算必须研究的基本理论。

数值计算的研究内容随着计算机发展和应用范围的扩大而不断扩大,从数学类型看,它包括**数值逼近**、**数值微分与数值积分**、**数值代数**、**最优化方法**、**常微分方程数值解法**、**积分方程数值解法**、**偏微分方程数值解法**、**计算几何**、**计算概率统计**等。

数值逼近是研究函数的离散逼近,用适合于计算机上计算的简单函数(如多项式,连分式等)逼近复杂函数。主要包括插值法、最佳一致逼近、最佳平方逼近与最小二乘逼近。各种代数插值是连续系统离散化的基础,理论与实践表明,分段低次插值比高次插值具有更好的稳定性。由于样条函数具有良好的特性,已成为函数逼近的重要工具而被广泛应用,样条逼近是函数逼近的一个重要研究方向。数值逼近也是生成初等函数子程序的重要方法。

数值微分与数值积分是在数值逼近基础上发展起来的计算微分与积分的数值方法。数值微分由于稳定性问题计算效果不理想,特别是对高阶导数和多元函数偏导数的计算效果更差。80年代提出的新方法是利用公式的数值求导或用符号运算的解析求导。数值积分常用的方法是数值稳定的,各种方法的差异在于计算量大小和自适应性,反常积分与高振荡积分要采用特殊的数值积分法。

快速傅里叶变换(FFT)是用三角函数逼近周期函数的递推方法,它把 N 个点变换的计算量从 $O(N^2)$ 下降为 $O(N \log_2 N)$,当 N 很大,效果尤为明显。FFT算法为调和分析方法在谱分析、信号处理和图像处理等领域应用计算机开辟了道路,也开创了各种快速算法的先河。

数值代数是有限维离散型方程的数值求解,内容包括**线性代数方程组数值解法**、**矩阵特征值问题数值解法**、**一元方程求根**、**非线性代数方程组数值解法**等。各种连续系统的离散化,最终归结为有限维离散型方程的数值求解。这类方程的数值解法分为两类,一类是迭代法,主要是构造收敛快、计算复杂性好的算法。对线性方程,人们针对不同类型方程特点,构造了行之有效的各种迭代法。对非线性方程,以牛顿法为代表的各种牛模型迭代法已得到广泛应用,但由于这类方法只具有局部收敛性,因

此,近20年来,具有大范围收敛的同伦连续法、单纯同伦算法、区间迭代法和单调迭代法等成为该领域的重要研究课题。另一类是直接法,它只对线性方程和极特殊的非线性方程有效,直接法由于舍入误差影响,方法的稳定性和方程是否病态等问题受到极大关注。当前病态方程组及大型稀疏方程组数值解法仍然是重要的研究方向。特征值计算包括标准特征值、广义特征值和反特征值问题的计算,反特征值问题由于条件的不同,提法各异,目前发展迅速并有广泛应用。

最优化方法是数值计算的一个迅速发展的领域,它研究在结构比较复杂的集合上泛函的极值问题。首先是规划问题,包括线性规划、非线性规划和动态规划等问题,经济学中许多问题可归结为这类问题。线性规划是理论成熟、应用最广泛的部分,基本解法是单纯形法。1984年,N. Karmarker提出了一种多项式时间的新方法,引起广泛注意。该方法的计算复杂度为 $O(n^{3.5}L^2)$,其中 n 为变量数, L 为输入长度。此外,运筹学和对策论中提出的极小极大问题及相应的计算方法也是最优化计算的研究内容。在经济和工程中求解大型最优化问题,目前已有可供使用的通用和专用的计算软件。

常微分方程数值解法包括初值问题与边值问题两类,初值问题的基本算法是龙格-库塔法和阿当姆斯法。近20年来,刚性问题的数值解法及其稳定性理论的研究有很大进展。边值问题的数值解法主要有两种,一种是转化为初值问题的打靶法;另一种是直接离散化的差分方法和变分方法。积分方程数值方法是直接将数值积分公式转化为离散化方程求解。

偏微分方程数值解法主要研究方程的离散化和对离散方程的求解,以及有关适定性、收敛性、稳定性、误差和自适应性等理论问题,使用计算机进行科学与工程计算往往涉及到偏微分方程的数值解法,诸如天气预报、反应堆设计的核扩散研究、液体流动、超音速流及弹性力学、地球物理、地震勘探等。

偏微分方程数值解法有正演问题和反演问题两类。正演问题是给定方程和定解条件求偏微分方程的解,它主要是二维和三维的定常问题和非定常问题。微分方程的离散化主要有差分法和有限元法两大类。差分法以差商逼近导数,将微分方程离散为差分方程,这种方法简单,适用于任何类型的方程,同时便于机器实现。有限元方法基于等价的变分原理的形式,采取任意格网分片逼近的手段,把传统对立的差分方法与里茨-伽辽金变分方法有机地结合

起来,发展成为解定常问题的主导方法,并推广到非定常问题。有限元法具有几何上灵活适应的突出优点,特别适合于解决复杂性大的问题。目前已有比较成熟的通用和专用有限元法计算软件,这些软件在众多科技领域和工程设计中已得到普遍应用。反演问题是在给定方程的模式下,已知其解或解的某一部分,要反推求该方程的系数和边界的形式等,起着导果求因的作用。这类问题在医疗卫生、无损探测、遥感遥测、地震勘探、地球物理等领域大量存在,它们通常是不适定的、病态的,有其特殊的难点。反演问题数值解法是一个正在开拓的新领域。

计算几何是数值计算的新分支,它研究静态或动态的几何形体和视象的离散化,以及逼近与生成的计算方法,是计算机辅助几何设计(CAGD)这门技术的数学基础。计算几何在计算机制图、计算机辅助设计、计算机辅助制造、计算机动画、计算机视象、机器人技术等众多领域有重要的应用。

计算概率统计主要研究随机数据的统计分析计算和概率系统模型的随机模拟计算。前者包含多元统计分析计算、时间序列分析计算、马尔可夫链计算和数值滤波等。后者最常用的是蒙特卡罗法。这方面的计算是根据实际问题的概率统计模型,对试验观测数据或随机模拟数据进行统计分析处理,给出实际问题性质的统计描述、统计控制或统计预测的数值结果。目前已研制出了众多的计算概率统计软件供用户使用。

数值计算软件是数值计算研究的另一个重要方面,每个数值软件实现一个特定的计算方法的标准程序模块,由一个或几个标准子程序组成。由于数值计算软件经常为用户使用,因此,它的研制除了要吸取一般软件理论和技术外,还有它自身的特点。首先要对计算方法进行认真选择,考虑方法的适用性、准确性、稳定性、计算量和存储量等因素。其次每个数值计算软件必须经过全面严格测试,以确保其正确性和可靠性。第三,每个数值计算软件需有详尽的说明书,对软件的使用、适用范围及正确性给予确切说明。目前,已问世的数值计算软件涉及到数值计算各领域。随着数值计算方法的进一步发展和计算机的更广泛应用,数值软件包含的内容也将越来越广泛。

并行算法是数值计算的一个特别活跃的新领域,它随着并行计算机的发展而迅速发展。并行计算机包括单指令流多数据流计算机(SIMD)和多指令流多数据流计算机(MIMD)两类,与其相应的算

法分为同步并行算法和异步并行算法两类。20世纪60年代末到70年代初,由于解偏微分方程及大型线性方程组等科学计算的需要,人们开始设计阵列式与向量式的新型结构计算机,开始研究同步并行算法。

1972年,阵列式计算机ILIACN-IV在美国宇航局投入运行,TI-ASC在欧洲运转,此后向量计算机CDCSTAR-100和CRAY-1等相继于1974年和1976年在美国投入运转。从1972年到1981年这一时期,并行算法进入实践的阶段与具体并行机的相关性增强,并行算法复杂性分析与评估有了新的内容,并行数值计算软件开始研制,但集中在串行程序的向量化研究上。

1982年—1991年是并行算法研究的兴盛期,在这一阶段,MIMD超级计算机系统CRAY X-MP,HEP与CRAY-2等相继投入运行,各类MIMD计算机系统如雨后春笋,特别是大规模并行SIMD计算机CM-2及MIMD计算机CM-5相继于1987年和1991年问世,一些大型科学与工程问题,如来自航天、化学、天体物理、材料科学、数据检索、石油勘探、天气预报等广泛领域中的问题,都能应用这种大规模并行计算机来求解。这时期并行计算的特点是:同步并行算法日益成熟并逐步完善;SIMD计算机语言开始形成以Fortran 90为基础的标准框架;异步并行计算成了热门研究课题。

1992年以后,并行计算开始转入标准化时期,这时期出现了规模更大速度更快的并行机,1994年思维公司推出了它的第五代并行处理系统CM-5E,它的峰值运算速度达到每秒830亿次浮点运算,CRAY研究公司也于1994年宣布了世界上最先进的并行处理机CRAY T3D系统,该系统由2048个处理机组成,其峰值运算速度达每秒3072亿浮点运算,这些对并行算法引起了深刻的变革,出现了一些新的研究方向,如格子气方法、基于神经网络的并行算法以及演化算法等。而并行计算语言的标准化和并行虚拟机的研究为并行算法的研究开辟了道路。

(李庆扬 康立山)

shuzhi jisuan wucha fenxi

数值计算误差分析 (error analysis of numerical computation) 关于数值计算中误差的产生与传播以及如何分析与控制各种误差的方法与过程。

数据近似值与精确值之差是衡量数据可靠性和

精确度的重要方面。应用数值方法在计算机上求解实际问题时,由于模型、测量手段和计算工具等方面的限制,以及计算方法的差异,所得结果往往不是所考虑对象的准确值,而是近似值。

设近似值为 a , 准确值为 x , 则 $e = x - a$ 称为近似值 a 的误差。通常不可能得到准确值 x , 也就不能算出误差 e 的值, 而只能根据数据计算出误差 e 的绝对值的上界 ε , 即

$$|e| = |x - a| \leq \varepsilon$$

称 ε 为近似值 a 的绝对误差限或误差限, 而准确值 x 的范围为

$$a - \varepsilon \leq x \leq a + \varepsilon$$

工程技术上常将上式表示为 $x = a \pm \varepsilon$ 。

在许多情况下误差限的大小还不能完全刻画一个近似值的准确程度, 它还与准确值 x 的大小有关, 比值 e/x 称为近似值 a 的相对误差, 并用记号 e_r 表示, 即

$$e_r = \frac{e}{x} = \frac{x - a}{x}$$

实际计算中常取 $e_r \approx \frac{x - a}{a}$, 并把相对误差 e_r 的上界称为相对误差限, 记作 ε_r , 即 $|e_r| \leq \varepsilon_r$, 计算时取 $\varepsilon_r = \frac{\varepsilon}{|a|}$ 。

数值计算的误差分析旨在估计数据计算结果的误差限。误差按其来源可分为模型误差、观测误差、截断误差和舍入误差等。

模型误差 数值计算首先要建立数学模型, 它是对被描述的实际问题进行抽象、简化得到的, 因而是近似的, 由此产生的误差称为模型误差。

观测误差 数学模型中常包含某些参量, 比如长度、温度、密度等, 需要通过观测确定它们, 由此产生的误差称为观测误差。

截断误差 数学模型中的表达式往往很复杂, 为了在计算机上求解, 必须提供合适的数值计算方法, 这种方法是通过将原数学模型的表达式作某种近似而产生的, 其误差称为截断误差或方法误差。如用 $x - x^3/3!$ 近似 $\sin x$ 时, 截断误差为

$$\sin x - \left(x - \frac{x^3}{3!}\right) = \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

舍入误差 用计算机进行计算时, 由于机器字长有限, 原始数据及计算过程中产生的数据, 其位数超过规定位数时都要进行舍入, 这样产生的误差称为舍入误差。

在数值计算中与误差有密切联系的另一个概念

是有效数字。

有效数字 若一个数 x 的误差不大于该数某位数字的半个单位, 则从非零数字最左边第一个数字起, 到这一位数字止都是该数的有效数字, 其个数称为该数有效数字的位数。例如数

$$x = \pi = 3.1415926\cdots,$$

若取近似值 $a = 3.14$, 它有三位有效数字, $|x - 3.14| \leq \frac{1}{2} \times 10^{-2}$; 若取 $a = 3.1416$, 它有 5 位有效

数字, $|x - 3.1416| \leq \frac{1}{2} \times 10^{-4}$ 。一般地, 设 x 的近似值 $a = \pm 10^m \times 0.a_1 \cdots a_n$, 其中 $a_i (i = 1, \cdots, n)$ 是 0 到 9 的一个数字, $a_1 \neq 0$, m 为整数, 且 $|x - a| \leq \frac{1}{2} \times 10^{m-n}$, 则近似值 a 相对数 x 具有 n 位有效

数字。实际上, 一个数按四舍五入规则得到的规格化近似数, 其每一位数都是有效数字。显然, 有效数字位数与小数点位置无关, 有效位数越多绝对误差与相对误差就越小。

按有效数字概念, 以下数字是有区别的, 79.06, 79.0600, 前者为 4 位有效数字, 后者为 6 位有效数字。

数值计算的误差分析是针对已建立数学模型的数值计算方法进行的, 它是一个重要而复杂的问题, 如果方法是近似的, 就需要进行截断误差分析, 不同方法有不同的截断误差估计公式。由于原始数据有误差, 而每一步运算又会产生新的舍入误差, 并传播前面各步已引入的误差, 所以逐步分析误差是可行的。误差分析是估计整个计算过程积累误差的界, 以判断数值计算结果的可靠性, 这是一个很复杂的问题, 目前尚无统一的方法, 常用的误差分析方法有以下几种:

向前误差分析法与向后误差分析法 假设所讨论的方法由某个公式表达, 某个结果 x 由已知量 (原始数据或先前已算出的量) a_1, a_2, \cdots, a_n 经过基本算术运算来确定, 可写成

$$x = g(a_1, \cdots, a_n)$$

由于计算中产生舍入误差, 实际计算出的值 a 与它的准确值 x 不同, 向前误差分析是根据误差运算规则对每一步运算找出舍入误差界, 随计算过程逐步向前分析, 直至估出最后舍入误差 $|x - a|$ 的上界, 这种误差分析方法只能应用于十分简单的情形。

向后误差分析则是把舍入误差与导出 a 的已知量 a_1, a_2, \cdots, a_n 的某种摄动 (微小误差) 联系起来,

即对某个 a_i 引进摄动量 ε_i , 使得由浮点运算得到等式

$$a = g(a_1 + \varepsilon_1, \dots, a_n + \varepsilon_n)$$

并推出这些 ε_i 的界 (ε_i 不是惟一的, 且无须求出 ε_i 的具体值), 然后利用摄动理论估计最后舍入误差 $|x - a|$ 的界。向后误差分析是一种先验性估计, 60 年代初, J H Wilkinson 在研究矩阵计算的误差分析时有较系统的讨论, 并取得了较大进展, 奠定了向后误差分析的基础。它是计算机上各种数值计算最常用的误差分析手段, J H Wilkinson 在他的著作“代数过程的舍入误差”^[2] 和“代数特征值问题”^[3] 中, 对数值计算中浮点运算的误差界及矩阵运算和数值代数求解舍入误差分析有详细的讨论。

区间分析法 60 年代中期提出了一种利用区间算法进行误差分析的方法, 它把参与运算的数 a, b, c, \dots 看成为分属于某些区间量 A, B, C, \dots , 通过对区间量运算求得结果 x 所属区间 X , 即求得近似值与误差。例如, 设 a^*, b^* 为准确值, a, b 是相应近似值, 由于 $|a^* - a| \leq \delta a, |b^* - b| \leq \delta b$, 则

$$a^* \in A = [a - \delta a, a + \delta a],$$

$$b^* \in B = [b - \delta b, b + \delta b]$$

利用 a^*, b^* 所在区间 A, B , 按区间运算规则可得到 a^*, b^* 运算结果 x 的区间 $X = [x, \bar{x}] = [c - \delta c, c + \delta c]$, 则 c 为 x 的近似值, δc 为误差限, 即 $|x - c| \leq \delta c$ 。这就是区间分析法的基本思想。

近 30 年来, 由区间运算发展起来的区间数学已形成一个新学科, 用区间方法求解数值计算问题, 得到的结果也是一个区间量, 它本身就包含了所求结果的近似值及其误差限。

概率分析法 通常对近似数运算结果误差限的估计总大于实际的误差, 实际上误差分布有随机性, 不会经常地达到上界。因此, 利用概率统计的方法, 将数据和运算中的误差视为适合某种分布的随机变量, 并由此推出计算结果的误差分布, 常常可以使误差估计更接近实际, 这种误差分布称为误差的概率界。

数值计算的误差分析目的是保证方法产生符合精度要求的可靠结果, 但对大量方法要定量分析舍入误差积累是非常复杂困难的, 上面提供的方法往往也是很难实现的。因此, 对舍入误差是否影响计算结果的可靠性进行定性分析是非常重要的, 这就是方法的数值稳定性问题。一个方法如果在执行它的过程中, 舍入误差的积累不影响产生可靠的结果, 则称该方法是数值稳定的, 否则称为数值不稳定的。

不稳定方法是不能使用的, 判断方法数值稳定的一个准则是原始数据的微小变化只会引起最后结果的微小变化。实用上只要方法是数值稳定的, 就不必再对它的舍入误差进行定量分析。

参考文献

1. 关治, 陈景良. 数值计算方法. 北京: 清华大学出版社, 1990
2. Wilkinson J H. Rounding Error in Algebraic Processes. Prentice-Hall, 1963
3. Wilkinson J H 著. 代数特征值问题. 石钟慈, 邓建新译. 北京: 科学出版社, 1987

(李庆扬 周树荃)

shuzhi weifen

数值微分 (numerical differentiation) 关于如何根据函数在一些离散点的值来推算它在某点的导数的近似值的方法。通常用差商来代替微商, 或用某个能近似代替该函数的较简单的函数 (如多项式函数、样条函数) 的相应导数作为所求导数的近似值。在微分方程中常用数值微分将方程离散化以求数值解。

通常采用多项式插值来求数值微分。设函数 f 在 $n+1$ 个等距点 $x_v = a + v h (v = 0, 1, \dots, n)$ 上的值 $f_v = f(x_v)$ 为已知, 通过低次插值可导出一些最基本和最常用的数值微分公式。例如, 两点公式 $f'(x_i) = \frac{f_{i+1} - f_i}{h} + O(h) = \frac{f_i - f_{i-1}}{h} + O(h) = \frac{f_{i+1} - f_{i-1}}{2h} + O(h^2)$, 三点公式 $f''(x_i) = \frac{f_{i+1} - 2f_i + f_{i-1}}{h^2} + O(h^2)$

等等。此外, 根据具有 $n+1$ 个等距结点的拉格朗日插值公式, 还可导出在结点 $x_i (i = 0, 1, \dots, n)$ 上的较为一般的数值微分公式

$$f'(x_i) = \frac{1}{h} \sum_{v=0}^n A_{i,v} f_v; \quad f''(x_i) = \frac{1}{h^2} \sum_{v=0}^n B_{i,v} f_v$$

其中 $A_{i,v}, B_{i,v}$ 仅与 n, i, v 有关, 相应的截断误差分别为

$$R'(x_i) = C_i h^n f^{(n+1)}(\xi_i)$$

$$R''(x_i) = \begin{cases} 2C_i \left(\sum_{v=1}^i \frac{1}{v} - \sum_{v=1}^{n-i} \frac{1}{v} \right) h^{n-1} f^{(n+1)}(\eta_i) & (i \neq \frac{n}{2}) \\ 2C_i h^n f^{(n+2)}(\eta_i) / (n+2) & (i = \frac{n}{2}) \end{cases}$$

其中 $C_i = (-1)^i i! (n-i)! / (n+1)!; \xi_i, \eta_i \in$

$(a, a + nh)$ 。因此,当结点的个数 $n+1$ 固定时,间距 h 愈小,则截断误差也愈小。但是,系数绝对值之和 $\frac{1}{h} \sum_{r=0}^n |A_{i,r}|$ 与 $\frac{1}{h^2} \sum_{r=0}^n |B_{i,r}|$ 随 h 的变小而剧增,所以函数值 f_i 的舍入误差对近似导数的影响也随 h 的变小而剧增。因此, h 并非愈小愈好,而是要适中,这是数值微分不同于某些插值之处。如果函数 f 有很好的可微性,即存在绝对值不太大的较高阶导数,则宁取间距稍大而个数稍多的结点。当 f 在结点分布的整个区间上的可微性不太好时,采用样条插值来进行数值微分比采用多项式插值更适宜,只是计算量要大得多。

如果数据 f_i 具有不容忽视的随机误差,且对应的自变量分布甚密,那么就应用曲线拟合来代替上述函数插值,然后用拟合曲线的导数作为函数 f 的导数的近似值。这样求得的导数叫做磨光的导数。

参考文献

冯康等. 数值计算方法. 北京: 国防工业出版社, 1978 (沈毅)

shuzhi

数制 (number system) 用一组统一的符号和规则来表示数的方法。它涉及到记数制, 定点数和浮点数的表示, 数的原码、补码、反码表示及其相应的运算规则。

记数制

用若干个数码的组合来表示 1 个数, 每个数码的值与其在数中所处的位置有关。例如一位十进制数可以用 0, 1, 2, ..., 9 等 10 个数码中的任一个来表示, 一位二进制数可以用 0 或 1 来表示。每个数码处在不同的位置时所代表的数值不同, 每个数码所表示的数值等于该数码本身乘以 1 个与它所在位置有关的常数, 这个常数称为权。一个数码可以选用的值的个数就是相应的记数制的基数, 十进制数的基数为 10, 二进制数的基数为 2。

十进制数的值可用 1 组有序的数码或多项式来表示, 例如

$$586.13 = 5 \times 10^2 + 8 \times 10^1 + 6 \times 10^0 + 1 \times 10^{-1} + 3 \times 10^{-2}$$

式中每个数位的权分别为 10^2 , 10^1 , 10^0 , 10^{-1} 和 10^{-2} 。

其他进位制的表示方法雷同。

在计算机中, 如果要求基本电子元件具有 10 个

稳定状态来分别表示数码 0~9 是很困难的, 但要求它们具有两个稳定状态来分别表示 0 和 1 却是容易实现的, 而且二进制数的运算规则比十进制的简单, 因此在计算机中广泛采用二进制数。

二进制数可用一有序数码或多项式表示, 举例如下:

$$1011.01 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$$

式中每个数码的权分别为 2^3 , 2^2 , 2^1 , 2^0 , 2^{-1} 和 2^{-2} 。

在计算机中的数有时也用八进制或十六进制表示, 通常用 3 位二进制码来表示 1 位八进制数码, 用 4 位二进制码来表示 1 位十六进制数码。

十进制、二进制、八进制和十六进制数如表 1 所示。表中 A, B, ..., F 分别表示十进制数的 10, 11, ..., 15。

表 1 各种进位制数

十进制数	二进制数	八进制数	十六进制数
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

通常输入计算机的是十进制数, 用 4 位二进制数来表示 1 位十进制数, 称为 BCD 码或二-十进制码(二进制编码的十进制数), BCD 码是有权码, 从高位开始各位的权分别为 8, 4, 2, 1, 所以又叫 8421 码。对于一位十进制数还有余 3 码和格雷码等表示方式, 如表 2 所示。余 3 码和格雷码是无权码, 余 3 码由 8421 码加 0011 得来, 格雷码有多种形式, 它的编码规则是使相邻两数码之间只有一位不同, 表 2 中列出两种格雷码。

表2 十进制数编码

十进制数码	8421 码	余3 码	格雷码	
0	0000	0011	0000	0000
1	0001	0100	0001	0100
2	0010	0101	0011	0110
3	0011	0110	0010	0010
4	0100	0111	0110	1010
5	0101	1000	1110	1011
6	0110	1001	1010	0011
7	0111	1010	1011	0001
8	1000	1011	1001	1001
9	1001	1100	1000	1000

二-十进制数输入计算机后,通常转换成二进制数,然后按二进制运算规则进行运算,在结果输出前转换成十进制数,然后再输出。某些计算机设置有十进制运算指令,可直接对十进制数进行运算。

定点数表示

在处理机的运算部件中,定点数的小数点位置是固定的,而且是隐含的(或约定的)。通常将小数点位置固定在数值的最右端或最左端,前者称为定点整数,后者称为定点小数。如果最左位定义为符号位,则定点整数的小数点位于数值部分之后,定点小数的小数点位于符号位之后,数值部分之前。例如01001,当默认为定点整数时,其值为+1001;当默认为定点小数时,其值为+0.1001。符号位为0时表示正数,为1时表示负数,-1001的机内码可表示为11001。

在计算机中,有时也可将数定义为不带符号的数,此时最左位是数据的最高位。

由于用户的初始数据、中间结果或最后结果可能在很大范围内变动,如果机器用定点整数或定点小数来表示数,程序员不得不在运算的各个阶段预先引入比例因子,将数放大和缩小,这会带给程序员很多麻烦。如果比例因子的选择没有达到恰到好处的理想情况,则运算时数据不是容易溢出就是容易丢失精度,因此某些处理机使用了小数点位置可以浮动的二进制浮点数表示形式。

浮点数表示

用浮点数表示的数分为阶码(或称为指数)和尾数两部分。阶码用二进制定点整数表示,尾数用二进制定点小数表示,阶码的长度决定数的范围,尾数的长度决定数的精度。例如+110100的数值等于 $2^6 \times 0.110100$,阶码为+110,尾数为+0.110100,

其浮点数的表示形式为:

0	0	1	1	0	1	1	0	1	0	0
尾数符号	阶码符号	阶码				尾数				

大多数计算机的浮点数表示符合 IEEE 754 标准,这时需对上述表示形式进行修正(参见浮点数标准)。

当尾数的绝对值大于或等于二分之一时,称为规格化浮点数。反之,就不是规格化数。考虑到尾数有正、负两种情况,如果尾数用补码表示,则可归纳如下:凡是尾数最高位与尾数符号位不同的数为规格化数,即尾数符号位为0,且尾数最高位为1;或尾数符号位为1,且尾数最高位为0的数为规格化数。在计算机内存储的浮点数一般为规格化数,在这种情况下,为了增加数据精度,可以将最高位规定为隐含位。计算机的阶码通常用补码或移码(增码)表示。

数的原码、补码和反码表示

以正负符号和绝对值来表示的数称为真值,计算机中的数称为机器数。在计算机中,分别用0和1来表示正号和负号。例如真值+0.1101,机器数也为0.1101;真值-0.1101,机器数为1.1101。在数的符号用0或1表示之后,根据运算需要,对数值部分可以有不同的处理方法,从而产生了原码、补码和反码3种表示方式。

原码 用最高位表示符号:0表示正数,1表示负数;有效数值部分用二进制绝对值表示,原码表示与上述机器数表示形式一致。

设真值 x 为定点小数,原码用 $x_0x_1x_2\cdots x_n$ 表示,且 x_0 为符号位,则

$$[x]_{\text{原}} = \begin{cases} x & 1 - 2^{-n} \geq x \geq 0 \\ 1 - x & 0 \geq x \geq -(1 - 2^{-n}) \end{cases}$$

设真值 x 为定点整数,原码用 $x_nx_{n-1}\cdots x_1x_0$ 表示,且 x_n 为符号位,则

$$[x]_{\text{原}} = \begin{cases} x & 2^n - 1 \geq x \geq 0 \\ 2^n - x & 0 \geq x \geq -(2^n - 1) \end{cases}$$

零值有两种:正零(00...0)和负零(100...0)。

如果数用原码表示,对两个数进行加减运算时,需要判定它们的符号和绝对值大小以后才能确定操作关系和操作类型,例如两个正数相减,大数应作为被减数;两个不同符号的数相减,应执行加操作,而且结果的符号还要进行单独处理。为了简化加减法

运算,可采用补码或反码来表示数。

补码 补码的正数与原码相同,负数则不同。补码的定义如下:

设真值 x 为定点小数,补码用 $x_0x_1x_2\cdots x_n$ 表示,且 x_0 为符号位,则

$$[x]_{\text{补}} = \begin{cases} x & 1 - 2^{-n} \geq x \geq 0 \\ 2 + x = 2 - |x| & 0 > x \geq -1 \end{cases} \pmod{2}$$

设真值 x 为定点整数,补码用 $x_nx_{n-1}\cdots x_1x_0$ 表示,且 x_n 为符号位,则

$$[x]_{\text{补}} = \begin{cases} x & 2^n - 1 \geq x \geq 0 \\ 2^{n+1} + x = 2^{n+1} - |x| & 0 > x \geq -2^n \end{cases} \pmod{2^{n+1}}$$

零只有 1 种表示方式,即为正零(000...0)。

定点数的补码运算具有以下特点:

$$[X+Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}} \pmod{2}$$

$$[X-Y]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}} \pmod{2}$$

反码 反码的正数与原码相同,负数则不同,反码的定义如下:

设真值 x 为定点小数,反码用 $x_0x_1x_2\cdots x_n$ 表示,且 x_0 为符号位,则

$$[x]_{\text{反}} = \begin{cases} x & 1 - 2^{-n} \geq x \geq 0 \\ (2 - 2^{-n}) + x & 0 \geq x \geq -(1 - 2^{-n}) \end{cases} \pmod{(2 - 2^{-n})}$$

设真值 x 为定点整数,反码用 $x_nx_{n-1}\cdots x_1x_0$ 表示,且 x_n 为符号位,则

$$[x]_{\text{反}} = \begin{cases} x & 2^{n-1} \geq x \geq 0 \\ (2^{n+1} - 1) + x & 0 \geq x \geq -(2^n - 1) \end{cases} \pmod{(2^{n+1} - 1)}$$

零有两种表示:正零(000...0)和负零(111...1)。

从以上定义可得出:

(1) 负数的反码的符号位与原码相同,但数值部分的每一位都相反,例如已知 x 为负数, $[x]_{\text{原}} = 1.0110$,则 $[x]_{\text{反}} = 1.1001$ 。

(2) 如果 x 为负数,比较 $[x]_{\text{反}}$ 和 $[x]_{\text{补}}$ 的公式,可得出:

当 x 为小数时, $[x]_{\text{补}} = [x]_{\text{反}} + 2^{-n}$; 当 x 为整数时, $[x]_{\text{补}} = [x]_{\text{反}} + 1$ 。即将 $[x]_{\text{反}}$ 的最低位加 1,可得 $[x]_{\text{补}}$ 。

浮点数的阶码常用移码(增码)来表示,移码的性质如下:

设阶码真值 x 为定点整数,其数值范围为

$-2^n \sim + (2^n - 1)$,移码用 $x_nx_{n+1}\cdots x_1x_0$ 表示,则

$$[x]_{\text{移}} = 2^n + x \quad 2^n > x \geq -2^n$$

将这一定义与定点整数补码的定义相比,可得出:

当 $0 \leq x \leq 2^n - 1$ 时, $[x]_{\text{移}} = 2^n + x = 2^n + [x]_{\text{补}}$;

当 $-2^n \leq x < 0$ 时, $[x]_{\text{移}} = 2^n + x = (2^{n+1} + x) - 2^n = [x]_{\text{补}} - 2^n$ 。

这表明,取 $[x]_{\text{补}}$ 的符号的相反值,保持其数值部分不变,即可得 $[x]_{\text{移}}$ 。例如:

$$x = +1011 \quad [x]_{\text{补}} = 01011 \quad [x]_{\text{移}} = 11011;$$

$$x = -1011 \quad [x]_{\text{补}} = 10101 \quad [x]_{\text{移}} = 00101。$$

零只有 1 种表示: $100\cdots 0$ 。

当阶码 $x = -2^n$ 时, $[x]_{\text{移}} = 000\cdots 0$,此为机器能表示的最小绝对值。一般当 $x \leq -2^n$ 时,不管尾数值为多少,一律以机器零表示,此时阶码与尾数均以全 0 表示。(王爱英)

shuzi cijilu

数字磁记录 (digital magnetic recording)

以正向或反向饱和磁化形式在移动的磁性薄膜介质上按磁道存储数据信息的技术。磁记录因其所记录的信号不同分为两种。一种是模拟磁记录,被记录的信号是模拟信号,如通常的录音、录像和记录供监测的振荡信号。另一种是数字磁记录,被记录的是经过编码的脉冲信号。记录媒体上呈现出一串饱和的磁化状态翻转,形成一条磁道。在移动速度恒定时,翻转间距的长短与一定频率的数字信号的编码方式相对应。

数字磁记录技术是磁表面存储器的基础,今天,它已发展成为由应用磁学、精密机械学、自动控制、半导体技术、计算机技术等多学科综合而成的一门在计算机领域中起重要作用的技术。

第一次出现磁记录是在 1898 年,丹麦人 V. Poulsen 发明了用单极型磁头在钢丝上录音的录音机。经历十余年,在三极管与放大器出现之后才得以实用化。1936 年,德国人 F. Pfeumer 用细粒磁粉涂布的磁带代替钢丝,使用环形磁头记录,改进了录音机。同时也开创了磁性材料与基底材料分离的先例,使磁性能与机械性能得以分别独立发展。

20 世纪 40 年代中期,计算机出现之后,使用磁鼓和磁带作存储器,数字磁记录设备成了计算机外存储器的主流,数字磁记录理论与技术获得了极大的发展,继而出现了速度快、记录密度高、存储容量

大的磁盘存储器。同时,与之密切相关的磁头、磁记录媒体、读写通路、磁记录编码、纠错码以及磁头定位与跟踪技术等也得到了同步发展。在全世界形成了以磁盘存储器为主体的庞大的磁记录工业。

特点 数字磁记录有以下特点。

(1) 数据的写入与读出速度快,因而数据传输速率高。

(2) 磁化状态翻转密度高。从原理上看,记录密度可以做到很高,目前,因受磁头和机械定位的约束,没有发挥具有的潜力。提高磁道密度和位密度还是今后追求的目标。

(3) 可方便地用交、直流抹除媒体上原有的信息;或在重写时覆盖,覆盖后原有信息的残余度很小。

(4) 能永久保留记录的信息,不致因掉电丢失。

(5) 读出信号为磁化媒体各层的集体贡献,不单纯取决于表层状态,故原始误码率较低,因而降低了数据检纠错的难度。

(6) 原材料供应丰富,价格低廉。

上述特点决定了磁记录技术在信息存储技术中处于主导地位,但该技术在使用中对环境的要求较高,要求防尘、防振、防止强磁场干扰。

记录过程 磁记录包含写入过程、读出过程和抹除过程。

写入过程 将待记录的二进制数经编码和电流放大后送入磁头线圈,在磁头铁心的前隙两侧产生磁压降,形成缝隙磁场。利用这个密集的磁场磁化具有永磁性质的磁层,使反映二进制数的写电流转换为可永久保留的磁化状态翻转图形,从而完成写入过程。由于线圈中流过的是极性按编码变换的电流,写入过程中媒体匀速移动,所以形成了由极性翻转的磁化单元所组成的磁迹,称之为磁道。磁道上N极与S极间的最小磁化状态翻转间距等于编码记录序列中两个1的相应间隔,如图1所示。



图1 纵向记录磁化翻转图形

读出过程 将媒体上的磁化图形还原为二进制数的过程称作读出过程,图2为以环形磁头读出时读出过程的示意图,当有剩磁的磁化图形通过磁头

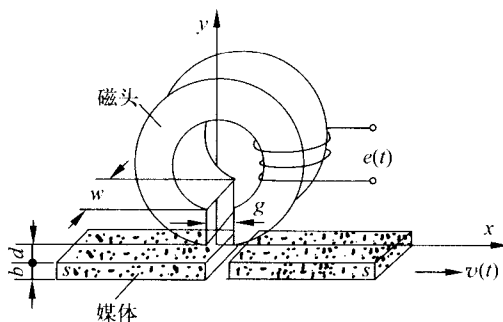


图2 读出过程

下方时,磁头线圈通过铁心与剩磁交链,变化的磁通在线圈中感应出电压信号 $e(t)$ 。在磁化翻转中心处,读出电压出现正、负峰值,两次翻转中间位置的信号幅度为零。若以彼此极性反向排列的永久磁铁模拟磁化翻转图形,用磁头在相对匀速运动中读取其剩磁,则从示波器上显示的读出波形如图3所示。此波形不是正弦波,而是与磁头磁场有关的特殊波形。它有如下的性质。

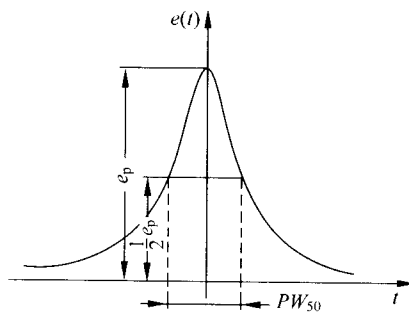


图3 读出信号波形

(1) 读出信号幅度 e_p 与媒体移动速度 $v(t)$ 成正比。当 $v(t)$ 是常数 v ,即速度恒定时,可写成 $e_p = kv$,其中 k 为比例系数。

(2) 图3中曲线 $e(t)$ 与 t 轴间所包含的面积是常数。即不管 e_p 幅度如何变化, $\int_{-\infty}^{+\infty} e(t) dt$ 是一个常数。由此可见,当 e_p 大时,波形是瘦高形的,而当 e_p 小时,波形是矮胖形的。通常用 $(1/2)e_p$ 处的波形宽度(称为 PW_{50})来表示波形形状特征。

(3) 按照法拉第定律,有

$$e(t) = -N \frac{d\Phi}{dt} = -N \frac{d\Phi}{dx} \frac{dx}{dt} = -N v \frac{d\Phi}{dx}$$

式中, N 为线圈匝数; $\frac{d\Phi}{dx}$ 为媒体上剩余磁通沿 x 方向的变化率, 它与媒体上的磁化状态有关。对于一根磁棒, $\frac{d\Phi}{dx}$ 最大值出现在它的顶部 (相当于磁记录中磁化状态翻转中心), $\frac{d\Phi}{dx} = 0$ 出现在它的中点 (相当于磁记录中的磁化单元的中间位置)。也就是说, $e(t)$ 的最大值 e_p 出现在磁化状态翻转中心通过磁头的时候。

(4) 信号幅度与磁头前极面至媒体表面间的距离 d 有关。距离缩小则信号幅度增加, 反之则衰减。对于软基底的记录媒体, 如磁带、软磁盘、磁卡片等, 多采用接触式记录, d 几乎等于零。对于硬基底的记录媒体, 如硬磁盘、磁鼓等, 多采用非接触式记录, 但 d 很小, 如硬磁盘驱动器中, 现在已达到 $0.1 \mu\text{m}$ 以下。

(5) 信号幅度与磁头在媒体上覆盖的宽度 w 成正比。

(6) 磁化状态翻转密度与读出信号波形的半幅宽度 PW_{50} 有关。显然 PW_{50} 小, 信号波形窄, 磁化状态翻转密度就高。从理论分析推导可以得到

$$PW_{50} = 2 \sqrt{(d+a)^2 + (g/2)^2}$$

式中, d 是磁头与媒体间的距离; g 为磁头前隙宽度; a 为磁化翻转过渡区参数, 它与媒体厚度 b 有关。

半幅宽度可用以估测媒体和磁记录系统的性能, 它与磁化状态翻转密度有如下的近似关系:

$$\frac{1}{PW_{50}} \approx C_{-3\text{dB}}$$

或

$$\frac{1.38}{PW_{50}} \approx C_{-6\text{dB}}$$

$C_{-3\text{dB}}$ 和 $C_{-6\text{dB}}$ 分别指读出幅度允许衰减 3dB 和 6dB 时的记录密度。与磁变阻 (MR) 磁头读出的过程类似。

抹除过程 它是将已记录的信息或残余的剩磁抹除的过程。采用交流磁场的抹除是使媒体磁性粒子无序排列, 磁媒体不再显示其磁性。而直流抹除则是使磁性粒子按同一方向排列, 不出现磁化方向翻转, 同样也读不出任何信号。通常, 在磁记录中使用的是重写覆盖, 无需先行抹除操作, 只有为防止道间串扰才使用磁道边缘抹除。

注意事项 读写过程中尚需注意以下问题。

波形拥挤效应 前述读出过程中所讨论的只是单次磁化状态翻转的读出波形, 忽略了相邻磁化状

态翻转读出波形的影响。任何数字磁记录的读出信号都是一连串的磁记录特殊波形, 彼此间相互影响。当磁化状态翻转密度很高时, 读出信号波形拥挤现象尤为显著。波形的拥挤导致幅度衰减与峰点偏移, 它将影响到数据与时钟的分离, 严重时造成数码错误。

幅度衰减主要因相邻波形极性相反相互削弱所造成, 峰点偏移则因两侧的增强与削弱不平衡而引起。例如对于码字为 11100111 的读出波形, 靠近二个 0 的 1, 其峰点将向 0 位偏移。所以针对不同的编码, 在写入时要进行补偿, 即提前或滞后写入。或者, 在读出时进行读后均衡, 以抑制峰点的偏移。

波形不对称 波形不对称主要由磁化的垂直分量引起。波形是由水平分量 (或称纵向分量) 和垂直分量两部分合成的。水平分量对称于纵轴, 而垂直分量则近似地对称于原点, 而不对称于两个坐标轴, 故合成后的波形不具备对称性。对于水平磁记录 (或称纵向磁记录), 由于媒体层厚极薄, 磁化的垂直分量很小, 故读出波形的垂直分量亦小, 波形的不对称性并不严重。对于垂直磁记录, 当以环形磁头读写时, 垂直分量较大, 读出波形具有明显的不对称性。为消除此种影响, 可以采用电路纠正的方法或在磁头结构上采取措施达到读出波形对称的目的。

类型

纵向磁记录 采用水平磁化场对平面内沿磁道方向取向的媒体进行磁化的磁记录称为纵向磁记录。目前, 绝大部分磁面存储器都采用纵向记录, 它使用环形磁头 (或环形磁头写入、磁变阻磁头读出) 与平面内取向的薄膜媒体。由于环形头的磁头场主要是水平分量, 媒体极薄, 且为平面内取向, 垂直于媒体表面的退磁场很大, 故记录后读出波形的对称性很好。它的翻转密度受各种因素的影响, 可使用翻转过渡区参数来衡量。理论上, 过渡区参数 a 可表示为

$$a = 2b(2 - S)M_r/H_c$$

式中, b 为媒体厚度, S 为媒体磁滞回线的矩形比, M_r 是媒体剩余磁化强度, H_c 是媒体的矫顽力。若要提高翻转密度, 便需减小过渡区参数 a 。从上式可知, 需要减小磁层厚度, 提高媒体材料的矩形比和减小比值 M_r/H_c 。

垂直磁记录 采用易轴垂直于表面的各向异性媒体和垂直磁场进行记录的方式称为垂直磁记录。与前述的纵向磁记录相比, 它有如下的特点。

(1) 记录后,磁化单元垂直媒体表面(如图4所示),且呈相互吸引结构,磁化状态稳定,翻转密度很高。

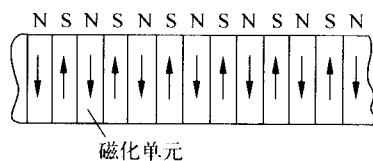


图4 垂直记录磁化翻转图形

(2) 适宜于用单极型磁头读写,但也能用环形头记录。当使用环形头记录时,波形严重不对称,必须进行特殊处理。无论使用何种磁头,都要求磁头与媒体表面间的距离极小,一般认为应小于 $0.1 \mu\text{m}$,甚至接触。

(3) 使用垂直表面取向的媒体,这种媒体有溅射 CoCr/Ni-Fe 的双层薄膜和涂布的钕铁氧体膜等。

垂直磁记录技术尚在实验研究中,除准垂直记录软磁盘外,硬盘的商品化还有一段距离。预期在不久的将来,该技术将成为提高磁记录密度的强有力手段。

参考文献

1. 张江陵,季国钧等. 外部设备设计原理. 武汉: 华中理工大学出版社, 1989
2. 杨正. 磁记录物理. 兰州: 兰州大学出版社, 1986 (张江陵)

shuzi cijilu bianma fangfa

数字磁记录编码方法 (encoding methods of digital magnetic recording) 按照数字磁记录的要求,为提高记录密度和可靠性而进行的代码变换。数字磁记录系统和数据通信系统类似,为实现某种目的,将信号按一定要求变换成二进制码序列。该变换过程称为编码。

编码 按功能不同可分为三类。

信源编码 将信源输出信号变换成二进制序列。目的是提高信息传输和存储的有效性。

检纠错编码 在信息传输或存储过程中,某些位出现由1变0或相反的差错时,利用冗余信息可以反映出错误图样与位置,可以自动检测或纠正。这种编码的目的是提高数据传输和存储的可靠性,即增强抗干扰能力。在通信系统中常称检纠错编码为信道编码。

记录编码 在数字磁记录系统中,为了提高记录媒体上的信息记录密度和可靠性,将原始数据序列变换成满足一定约束条件的记录序列的变换过程称作记录编码,也叫记录方式或调制方式。

码型 记录编码的具体形式。码型有多种类型,如图1所示。按工作原理不同可分为三类:未编码直接记录、游程长度受限码(RLLC)和加扰码(或称随机化不归零码)。

未编码直接记录 即**不归零制(NRZ)**码,电报码是其中一种。它按数据序列直接记录,在磁层上以

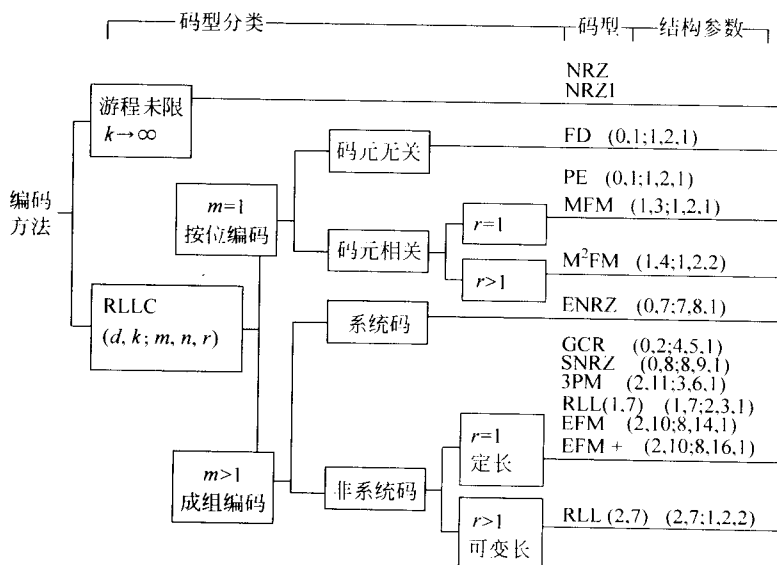


图1 数字磁记录编码分类

剩磁 $-B_r$ 表示 0, $+B_r$ 表示 1, 对序列中 0, 1 的排列未作任何变化。NRZ 的特点是: 仅当从 0 变 1 或从 1 变 0 时才能检测到读出信号; 跟在 0 后面的头一个 1 可检测到一个正信号脉冲, 而在它后面的所有的连续 1 均无信号脉冲。需要特殊标记表征记录的开始。它的读出信号中没有自同步能力, 若有一位出错, 则其后续一串信息位均错, 直到下一个信号脉冲为止。这种码型属于误码传播的码型, 它无法区分存储位无信号或是媒体发生信号丢失这两种情况。改进后的不归零制码称作**逢 1 变化不归零制**(NRZI)码。它的记录规则是: 对应数据序列中的 1, 磁层上磁化状态翻转一次, 若为 0 则维持原磁化状态不变。NRZI 的特点是: 仅当存储数据 1 时, 读出才有脉冲信号(正或负脉冲), 存 0 时无读出脉冲。在某一位上读出的错误只限于该位, 不影响后继位。NRZI 属于误码传播受限码, 可靠性优于 NRZ。尽管从其频谱角度分析, 两者相同, 但因 NRZI 具有信息独立性而成为数字磁记录系统中最基本的码型。

游程长度受限码 为实现高密度记录, 既能避免读出脉冲拥挤又能提取自同步脉冲的一种编码。连续 l 个 1 或 0 构成的数据串, 称作长度为 l 的游程。对记录序列中 1 和 0 的游程长度均作限制的一类信道编码, 称作游程长度受限码(RLLC)。在 RLLC 序列中, 两个相邻 1 之间至少要有 d 个 0, 最多为 k 个 0 (易于提取自同步脉冲的条件)。其中, d, k 均为正整数, 称为约束参数。因此, RLLC 码又称 d, k 受限码。20 世纪 70 年代形成的 RLLC 理论能对**数字磁记录**和**光记录**系统的主要实用码型进行统一的数学描述和理论概括。和前述的信源编码、检纠错编码不同, RLLC 实质上是一种特殊的码型变换, 类似于二进制与十进制之间的相互变换, 但 RLLC 变换的“权”不是 2^j 或 10^j (其中 j 为正整数), 而是特殊的数列。当 $k=1$ 时, 它是斐波那契数列; $k>1$ 时, 为广义斐波那契数列。RLLC 理论还能指导记录编码的设计和工程实现。RLLC 的统一数学描述较严谨、抽象。实践中常用 $(d, k; m, n, r)$ 表示, 以便将各种码型进行分类并对其主要性能作定量评价。其中, d, k 为约束参数, d 是用来限制 1 游程的, 为避免读出脉冲拥挤, k 是限制 0 游程的, 为便于提取自同步脉冲; m 表示输入数据序列的位数, $m \geq 1$; n 表示变换成 d, k 受限的记录序列后的位数, 因为要删除一些不符合约束条件的序列(非法码字), 显然 $n \geq m$ 及 $2^n \geq 2^m$ 是选取 n 的

必要条件; r 是变换参数, 即变换过程中数据串的最大长度与最小长度的比值。利用上述 5 个结构参数, 可将码型分类: $m=1$ 的一类称为按位编码, $m>1$ 的一类称为成组编码; $r=1$ 表示分组长度固定, $r>1$ 表示分组长度可变。

RLLC 的构造(编译码过程)与其结构参数密切相关。除未经编码的 NRZ, NRZI 以及加扰码(包括 Scramble NRZ, Randomized NRZ)外, 绝大多数早期及现今实用码型均属 RLLC。若将 NRZ, NRZI 用 RLLC 结构参数表示, 它们均为 $(0, \infty; 1, 1, 1)$ 码。

下面是 RLLC 的一些码型。

相位编码(PE)也称调相制(PM), 或威廉码, 或双相码, 在计算机中称为曼彻斯特码。其规则是: 记录 0 时, 磁化状态由负电平(或负脉冲)变到正电平(或正脉冲); 记录 1 时, 磁化状态由正变负, 两者相位差 180° 。

倍频码(FD)也称调频制(FM), 或哈佛码。其规则是: 记录 1 时, 在位单元前沿和中央处磁化状态各翻转一次; 记录 0 时, 只在位单元前沿处翻转一次。1 的读出波形频率是 0 波形频率的两倍。PE 和 FD 均属 RLLC, 可表示为 $(0, 1; 1, 2, 1)$, 其中 $m=1, n=2$, 一位输入数据变换成二位记录序列 (e_1, e_2) ; $d=0$, 允许记录序列中出现连续 1; $k=1$, 表示不允许记录序列中出现连续 0。

改进调频制码(MFM)具有 FD 码的自同步能力, 又具有编码效率为 100% 的优点, 故称改进型。这类码又称密勒码, 或窄带码。编码规则是: 记录 1 时, 在位单元中央磁化状态翻转; 记录单个 0 时, 磁化状态维持原来状态不变; 当记录连续 0 时, 在两个 0 的交界处磁化状态翻转。它也是一种 RLLC, 可用 $(1, 3; 1, 2, 1)$ 表示。从时域波形图分析, 三种频率分别对应磁化状态翻转间距 $T, 1.5T, 2T$ (T 为记录序列的码元周期)。从频域分析, 该码型功率谱密度分布很集中, 占据很窄的频带, 因此, MFM 抗干扰能力强。

表 1 列出 PE, FD, MFM 的编译码规则。其中 d_{-1} 表示当前待变换数据 d_0 的前一位, PE 和 MFM 的记录序列 $e_1 e_2$ 取值均和 d_{-1} 状态有关。译码时要判读 d_0 是 0 还是 1 时也要依据 d_{-1} 的数值。这三种码型属于码元相关一类, 信息独立性差, 误码会传播。PE 用于早期磁带机, MFM 广泛用于 20 世纪 70 年代的各种磁盘机、高密度数字磁带记录仪, FD 用于早期磁鼓、磁带机及软、硬磁盘机。

为了形象地说明上述三种按位编码的 RLLC 的

磁化状态,将它们的波形图与 NRZ, NRZI 的波形图均示于图 2 中,以供对比。

表 1 不同码型的编译码规则

码型	数据序列 d_0	记录序列 $e_1 \ e_2$	条件 d_{-1}
PE (PM)		1 1	0
	0	0 1	1
		0 1	0
	1	1 1	1
FD (FM)	0	1 0	无
	1	1 1	无
MFM	0	1 0	
	1	0 1	0
	0	0 0	
	1	0 1	1

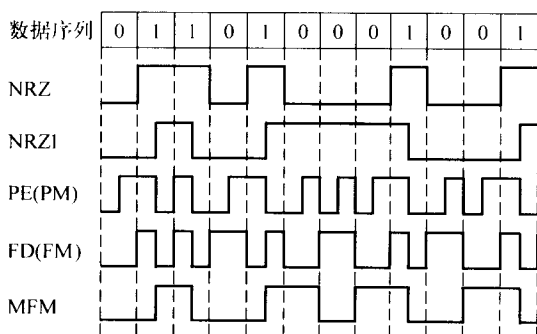


图 2 几种码型的波形图

成组编码(GCR)是 RLLC 中 $m > 1$ 的一类编码。按变换参数 r 取值又分为两类: $r = 1$ 为定变比即定长编码; $r > 1$ 为可变长编码。

$r = 1$ 这类码型有: ① GCR(4/5), (0, 2; 4, 5, 1) 码, 它曾广泛用于倍密度软盘及 6 250 bpi 磁带机; ② 三单元调制(3PM), (2, 11; 3, 6, 1) 码, 曾用于高密度磁盘机、高密度数字记录仪。③ 增强不归零(ENRZ), (0, 7; 7, 8, 1) 码; ④ 同步不归零(SNRZ), (0, 8; 8, 9, 1) 码; ⑤ 只读光盘(CD-ROM)中采用的 EFM(2, 10; 8, 14, 1) 码; 一写多读光盘(WORM)中采用的 RLL(4, 15)及 DVD 采用的 EFM plus(2, 10; 8, 16, 1) 码均属于此类。磁光盘及高密度磁盘均采用 RLL(1, 7)即(1, 7; 2, 3, 1)码。

$r > 1$ 这类码型中以 $r = 2$ 较为实用, 如 RLL(2, 7)即(2, 7; 2, 3, 2) 码。它广泛用于高密度磁盘、流式磁带机、130 mm 磁光盘及部分 WORM 产品中。

加扰码 也称随机化不归零制。加扰码和 RLLC 概念不同, 不是简单地变换码制, 而是用伪随机序列产生器将给定的输入数据序列加扰成新的伪随机序列, 使其中 0 和 1 的分布概率相等, 以便提取自同步时钟脉冲。其编码效率为 100%, 变换后的记录序列仍按 NRZI 规则记录在磁层上。加扰的实质是选择一个合适的本原多项式, 对输入数据序列进行运算, 使其输出序列具备伪随机性质。解扰过程相当于译码, 是加扰的逆过程。该码型主要用于高密度磁带记录仪, 在 CD-ROM 光盘标准中也利用了这种加扰概念。

码型评价准则 选用和评价众多码型时有以下一些主要准则。

检读窗宽 T_w 它是一个时间量, 表示判读 1 或 0 的允许时间值。令 T_0 表示数据序列的位周期, T 表示变换后记录序列的位周期。因为记录系统的数据传输速率不变, 所以 $mT_0 = nT$, 最大允许检读窗宽 $T_w = T = (m/n)T_0$ 。

编码效率 M 也称密度比, 是数据记录位密度与最大磁化状态翻转密度的比值。实质上指每次磁化状态翻转所能存储的数据位, 规定以 NRZI 的编码效率 $M = 100\%$ 为基准。 M 值越大表示存储相同数据位时磁层上状态翻转次数越少。选码型时尽可能选择 M 值大的, 以使读出信号幅度大, 信噪比大, 抗干扰能力强。这样, 在相同位密度记录时, 对磁头、磁层、电子线路的高频响应可降低要求。 $M = B/F = (1/T_0) / (1/T_{\min}) = T_{\min}/T_0 = m/n(d+1)$ 。其中, B 是数据密度, 单位是 b/mm; F 表示每毫米长度磁层上磁化状态翻转次数的最大值。

带宽 不同码型的磁化状态翻转规律在频域上表现为不同的最高和最低重复频率, 即它们的带宽不同。当记录信道带宽受限时, 要求码型高频受限且能避开直流或甚低频区域。用参数 P 表示最大磁化状态翻转间距(T_{\max})与最小磁化状态翻转间距(T_{\min})之比, 即 $P = T_{\max}/T_{\min}$, 因为 $T_{\max} = (k+1)T = (k+1)(m/n)T_0$, $T_{\min} = (d+1)(m/n)T_0$, 所以 $P = (k+1)/(d+1)$ 。 P 值小表示窄带码型, 抗干扰能力强, 易于自同步, 对信号选择性好, 电路容易实现。自同步能力指从单个磁道读出脉冲序列中提取同步时钟脉冲的难易程度。时钟脉冲是判读每位是 0 或 1 的时间依据。

编码比 R 亦称变换比或码比。 $R = m/n$, 由于 $m \leq n$, 因此 $R \leq 1$ 。若 $R = 1$ 表示信息容量无损失。希望选择 R 接近 1 的码型。

数字和变化量(DSV) 反映信道低频段畸变的程度,又称为“编码电平”变化量。在时域的波形图上表现为横轴基线上的上抬或下降。若1,0分别用+1,-1电平表示,则DSV的物理意义相当于编码记录序列通过一个电容器之后,在其两端上积累的电荷最大差值。高密度记录码型的DSV是个有限数值,表示直流分量没有或很小,NRZ和NRZI的DSV可能趋于无限大。

编译码规则 编译码规则要简单,用硬、软件容易实现,经济而易于维护。

评价和选用记录编码的方法有许多种。主要依据时间域、频率域、磁头与记录媒体的耦合特性及经济实用等因素来选择。其中,磁头与媒体的耦合特性是信道的固有特性,是选用最佳记录码型的最重要因素之一。对码型性能的实际测试可借助数据通信中测试传输码的眼图观察评定法,也可以针对码型选用或设计测试码,或者制作专用测试仪器。

新的记录编码

(0,G/I)序列 RLLC码概念的扩展。主要表现在磁记录系统引入部分响应最大似然判读(PRML)之后,一些新的码型获得应用。如20世纪90年代后期硬盘机中采用的8/9(0,4/4)RLL码及后继的约束条件放宽,码比更高的16/17(0,6/6)RLL码等都属于(0,G/I)序列。

在(0,G/I)序列中,0可视为 $d=0$ 的约束,而G为连续0的总数目,即两个相邻1之间所允许的0的最大数值。在记录信道输出端每一次奇数及偶数交错的0的游程长度规定不超过一个特定的正整数I。G类似于一般RLLC中的k,该序列中的0,即 $d=0$,强调在PRML系统中码间干扰是可以承受的。(0,G/I)和交错NRZ预编码器的组合已用来防止EPR4信道中近乎灾难性的误码传播。

(d,k,c) charge-RLL 另一种码型,其中c表示附加约束,即NRZI双极性的记录序列中无直流分量,其DSV小于等于某一数值N,而 $N=2c+1$,即 $c=(N-1)/2$ 。无直流分量序列已广泛用于旋转磁头DAT(R-DAT)磁带机中防止因变压器耦合引起写信号畸变;在光盘为减少数据与伺服信号之间干扰以及可能因盘片污点引起的低频噪声均采用DC-free码。

格构码 一种高密度、高可靠编码。早已发现磁记录系统中绝大多数普通误码均源于连续检测到三次或更多次磁通翻转失败。引进 (d,k) 约束增大

码间距离,为检测或纠正误码提供了条件。这种编码限制了最大翻转次数,称为MTR码。这种编码本身具备结构性冗余信息,附加一套与冗余信息匹配的检读电路,称为格构编码系统,简称格构码。

二维RLL码 以上介绍的均是沿磁道方向提高记录密度的编码,减少磁道宽度并提高道密度从而提高面记录密度是有潜力的。为避免道间干扰和提高信噪比所用的二维RLL码可表述为 $(dx, kx, dy, ky; n)$,其中 dx, kx 同单道RLL码, dy 是相邻磁道间最小游程, ky 是全部n磁道中最大游程。实验证明:同一设备的二维RLL码的容量高于单道RLL码。

参考文献

1. 郭平欣,姚锡珊,虞浦帆主编.电子计算机外部设备原理.北京:国防工业出版社,1986

2. Immink A S, Siegel P H, Wolf J K. Codes for digital recorders. IEEE Trans. on IT., 1998, 44(6)

(刘庭华)

shuzi cijilu jiancuoma

数字磁记录检错码(error detection codes of digital magnetic recording) 通过增加冗余信息以发现被记录信息因受噪声干扰产生的某些错码的一类码。它是纠错码的基础。设记录信息为k位,冗余信息即检验位为r位。实际记录序列长度 $n=k+r$ 。现有检错码均为线性码,检验规则可用线性方程组表示。实用检错码主要有奇偶检验码、重复码、水平垂直一致检验码(矩阵码)、等比(等重)码以及循环码等。

奇偶检验码 应用最广、最简单的一种检错码。它是在二进制数序列的末尾附加一位检验位,在记录序列(n位长)中选择最末位数字使该序列中的1的总数为奇(或偶)数,称为奇(或偶)检验码。无论按奇数或按偶数编成的码字均满足奇(或偶)检验的同一种规则,因此此类码又称为一致检验(监督)码。无论信息位k的长短,只要增加一位检验位即可构成。其编码效率 η 高, $\eta=(n-1)/n$,但检错能力很低,即只能检测奇数个错误,但不能确定其位置;不能发现偶数个错误。

水平垂直一致检验码 曾在多道磁带机中应用,它沿磁带运动方向(水平)及垂直磁道方向两个方向采用奇偶检验。这种检错码检错能力较高:①能检测全部奇数个错误;②绝大多数偶数个错误;③若在某一定位上,水平与垂直检验同时显示有错还可以纠正该位信息。

重复码 检验位是信息位的重复故取此名。早期磁带机中曾采用,该码效率过低,现已少用。

等比码 记录序列中 1 的个数固定,又称等重码,若序列中 1 与 0 的数量的比率不变则称为定比码。它是一种非线性码,可检测所有的非对称性错误,即 0 错为 1 或 1 错成 0 而导致的错误,或冒码与漏码错误。现在磁光盘(B)格式中采用的 4/15 码即定比码。

循环码 若码 C 中任一码字任意循环移位后仍为 C 中码字,即码 C 对循环移位具有封闭性,则称 C 为循环码。它是一种线性分组码,计算机数字磁记录系统及数字通信中采用的检纠错码多为循环码。其特点是:①码的结构容易用代数方法分析与构造;②具有封闭性,编译码过程可用线性移位寄存器实现且容易获得简捷的译码算法。

任一码字均可用伪变量 X 的多项式表示。二元域中,可将多项式的各项系数设为 1 或 0 来对应码字的各个码元。例如已知码字 110101,其多项式为 $1 \cdot x^5 + 1 \cdot x^4 + 0 \cdot x^3 + 1 \cdot x^2 + 0 \cdot x + 1 \cdot x^0$,即 $x^5 + x^4 + x^2 + 1$,若按升幂排列可表示为 $1 + x + x^3 + x^5$ 。

若 (n, k) 循环码的每个码字 $C(x)$ 都是一个惟一的、次数为 r 的、首尾项系数等于 1 ($G_1 = G_0 = 1$) 的码多项式 $G(x)$ 的倍式,则 $G(x)$ 是循环码的生成多项式。码字 $C(x)$ 最高次数是 $n-1$,信息多项式 $m(x)$ 的次数为 $k-1$, $C(x)$ 与 $G(x)$ 间有下列关系:

$$C(x) = m(x)G(x) = 0 \pmod{G(x)}$$

而 $G(x) = G_r x^r + G_{r-1} x^{r-1} + \dots + G_1 x + G_0$,可见循环码 (n, k) 的 $G(x)$ 一定是 $x^n + 1$ 的因式。所谓检纠错码实质上就是选择 $G(x)$ 构成符合检纠错要求的循环码。

码字间的距离表示两码字之间差别的大小。在有干扰的情况下,距离越大,则一码字变成另一码字的可能性越小。码字间的汉明距离定义为两个码字之间对应位上取值不同的数目。对于码的所有 2^k 个码字集合 C 中,任意两个码字之间距离的最小值称为最小距离,记作 d_{\min} 。检测出 L 个错误的充分必要条件是 $d_{\min} = L + 1$,或 $L = d_{\min} - 1$ 。 d_{\min} 表示检测能力,表明可检测出所有个数小于或等于 L 的独立随机错误,但不能检测出 $L + 1$ 个错误。循环码最适宜用以检错,主要检错能力归纳如下:①可检测任意奇数个错误;②能检测出全部突发长度 $b \leq (n - k)$ 的任何错误;③当 $b > n - k$ 时,有些模式的错误是不能检测出的,但比例很小;④能检测 $b \leq 2$ 的两个突发错误。

所谓突发错误是指错误之间具有相关性而不是彼此独立的一串错误。在磁表面存储设备中,因存在磁层脱落、机械损伤等,突发错误是主要的。

奇偶校验码、重复码、汉明码等检错主要针对随机错误,Abramson 码、RS 码、RS 乘积码等循环码可纠检随机与突发错误。

循环冗余检验码(CRC) 循环码的子类,其实用码型均属于 Abramson 码,生成多项式为 $G(x) = (x+1)G_1(x)$ 。例如,CRC-16 曾用于盒式磁带机及 QIC-24 等流式磁带机,其生成多项式 $G(x) = x^{16} + x^{15} + x^2 + 1 = (x+1)(x^{15} + x + 1)$;又如 CRC-CCITT 是国际电报电话咨询委员会(CCITT)推荐的检验码,其 $G(x) = x^{16} + x^{12} + x^5 + 1 = (x+1)(x^{15} + x^{14} + x^{13} + x^{12} + x^4 + x^3 + x^2 + x + 1)$ 。它用在软盘机控制器中作为通信控制检错码。这两种检错码的检错能力相同,可直接采用市售 CRC 芯片(例如 MC 8503)实现,不必设计编译码电路。(刘庭华)

shuzi cijilu jiucuo ma

数字磁记录纠错码(error correcting codes of digital magnetic recording) 具有对信息进行检测并纠正其错误能力的码。它是一类强抗干扰的信道编码,不仅能发现错误模式还能确定错误发生的位置。

分类

纠错码的分类见图 1。数字磁记录中主要采用线性分组码所属的各类纠错码。

线性码 检验位与信息位之间呈线性关系,检验规则可用线性方程组表示。

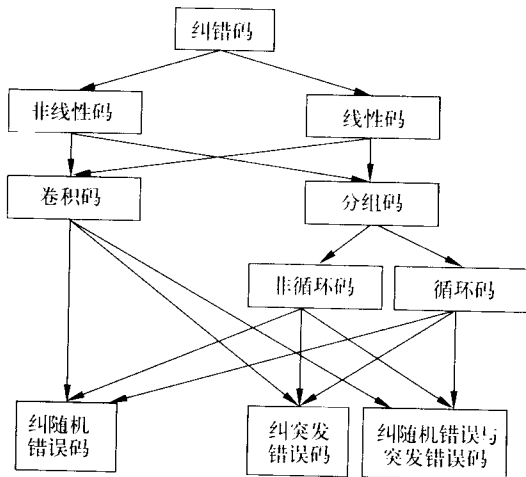


图 1 纠错码的分类

分组码 若码组长度为 n 位, 信息位为 k 位, 则每一码组的 $n-k$ 个检验位仅与本码组的 k 个信息位有关, 而与别组的信息位无关。

卷积码 本组的检验位不仅与本组信息位有关, 而且与前后若干组的信息位有关, 也称为连环码。

循环码 指同一组的所有码字均可由任一码字的循环移位获得。

非循环码 不能从循环移位获得全部码字。如磁盘机中曾广泛应用的缩短循环码。

纠随机错误码 码字之间的差异称为距离, 即两码字的对应位不同的数量。设码字间最小距离为 d_{\min} , 则纠随机错误码的纠错能力表述如下。

纠正 t 个错误的必要充分条件是 $d_{\min} \geq 2t + 1$, 即正整数 $t \leq (d_{\min} - 1)/2$ 。

纠正 t 个并发现 l 个 ($l \geq t$) 错误的必要充分条件是 $d_{\min} = t + l + 1$ 或 $(d_{\min} - 1)/2 = (t + l)/2 \geq t$ 。

所以纠错数目 $t \leq (d_{\min} - 1)/2$, 检错数目 $L \leq t + l = d_{\min} - 1$ 。

纠突发错误码 突发错误指误码之间存在相关性而非独立随机的, 此类纠错码是数字磁记录系统主要的实用码型之一。设码长 n 位, 信息位为 k 位, 检验位 $r = n - k$, 可纠突发长度 $b \leq (n - k)/2$, b 取整数上限值。通常增加 r 值来增强纠错能力, 如磁盘纠错用的法尔码。

纠随机错误与突发错误码 此类码既可纠独立随机错误又可纠错误相关的突发错误。例如, 给定一个 (n, k) 循环码, 可构成码长和信息位分别扩大 λ 倍的循环码, 将原来码中的 λ 个码字排成码阵中的 λ 行, 然后按列传送, 可构成交错码。参数 λ 称为交错次数或交错度, 即码阵中的行数。交错次数愈大, 错误相关性愈弱。因为长度为 λ 的突发错误无论从何处开始, 至少影响每一行中的一位。这种交错码的纠错能力如下: ①若原来码能纠单个随机错误, 则交错码可纠长度小于或等于 λ 的单个突发错误; ②若原来码可纠正小于或等于 t 个错误的各种组合, 则交错码可纠正 t 个长度小于或等于 λ 的突发错误的任意组合; ③若原来码可纠正任何长度小于或等于 b 的单个突发错误, 则交错码可纠正长度小于或等于 λb 的任意单个突发错误。

此外, 纠随机错误与纠突发错误码还有能纠正单个比特错误和单个长度为 b 的突发错误的最佳矩阵码 (ORC), 它用于 6 250 bpi, GCR (4/5) 磁带机中。

应用

早期磁带机边读出边检测, 曾采用汉明码, 能纠

正出现的各种单个随机错误, 扩展汉明码能纠一检二或检测三个错误, 发展到 IBM 3850 海量磁带存储系统时, 采用能纠正多个随机错误的 BCH (15, 13) 码。20 世纪 70 年代 6 250 bpi, GCR (4/5) 半英寸 (12.7mm) 9 道磁带机采用 ORC 码。早期磁盘机 IBM 1300, 2311 等均采用循环检验码 (CRC)。随着位密度的提高, 20 世纪 70 年代开始采用纠正单个突发错误的法尔码, 该码是由生成多项式 $g(x) = (x^c - 1)p(x)$ 所生成的 (n, k) 循环码。其中 $p(x)$ 为 m 次不可约多项式, 其周期是 e , 码长 $n = Lcm(e, c)$, $g(x)$ 的次数 $r = c + m$ 。其纠错能力如下: ①若 $c + 1 \geq b + d, m \geq b$, 则能纠正单个长度小于或等于 b 的突发错误, 且同时可检测任何长度大于或等于 b , 但小于或等于 d ($d \geq b$) 的突发错误。②若只作检测, 能检测长度 $b \leq r = c + m$ 的任意单个突发错误或检测长度分别为 d, b 的两个突发错误的任意组合。其中 $c + 1 \geq b + d > 2b, d \geq b, m \geq b$ 。美国 CDC 公司 Cyber 170 844 磁盘机曾采用 $g(x) = (x^{21} + 1)(x^{11} + x^2 + 1)$, 可纠 11 位突发错误的法尔码, IBM 3330 盘采用过纠 11 位的 56 次生成多项式的扩展法尔码, 我国 1001 型磁盘采用过 $g(x) = (x^{16} + 1)(x^8 + x^4 + x^3 + x^2 + 1)$, 纠 8 位的法尔码等。1978 年 HP 13037 磁盘采用检纠错能力更强, 编译码方法与法尔码类似的 Gilbert 码, 其生成多项式 $g(x) = [(x^8 + 1)(x^9 + 1)] / (x + 1)$, 码长 $n = p \cdot q, p, q$ 互素, 检验位数 $n - k = r = p + q - 1$, 信息位数 $k = (p - 1)(q - 1)$ 。它能纠正 32 位突发错误, 检测大于 32 位但不超过 48 位的突发错误。所用的 $p = 43, q = 53$, 检验位 95, $k = 2184, n = 2279$ 。20 世纪 80 年代后高档磁带机和磁盘机采用光盘系统中广泛应用的 RS 码。

参考文献

顾慰文. 纠错码及其在计算机系统中的应用. 北京: 国防工业出版社, 1980 (刘庭华)

shuzi duoyongtu guangpan

数字多用途光盘 (digital versatile disc, DVD) 在 CD 光盘基础上通过减短半导体激光器的波长 (从 780 nm 到 650 nm)、增大物镜的数值孔径 (从 0.45 到 0.60) 等技术手段, 从而提高存储密度的高密度光盘系统。它是 CD 光盘的更新换代产品。系统由 DVD 光盘机和 DVD 光盘两部分组成, 简称 DVD 光盘或 DVD。

DVD 光盘由美国时代华纳公司提出构想, 日本东芝公司研制, 开始称为超高密度数字视频光盘

(SD);接着,以荷兰飞利浦公司、日本索尼公司为首研制出多媒体数字光盘(MMCD),因而形成两大阵营。1995年12月,两大阵营在多次协商的基础上统一定名为DVD。1996年推出了DVD的第一个产品——DVD视频播放机。

分类

DVD光盘机根据用途的不同可以分为DVD光盘驱动器、DVD光盘播放机和DVD光盘录像机等。DVD光盘驱动器用作计算机外部设备,对DVD或CD光盘进行信息的读取、写入、擦除。光盘驱动器根据其使用的DVD或CD光盘类型不同,分为只读、一次写及可擦写三类光盘驱动器。DVD光盘播放机是只能回放DVD-Video、DVD-Audio等的信息家电产品。DVD光盘录像机既可以是一次写的,也可以是可擦写的,并具有播放机功能。

DVD光盘虽与CD光盘外型尺寸完全一致,但它是由二张0.6 mm厚的盘片粘接而成。DVD光盘根据功能不同可以分为只读、一次写、可擦写和复合四种。只读DVD光盘有DVD-ROM、DVD-Video和DVD-Audio,DVD-ROM用于存储计算机读取的信息。一次写DVD光盘有DVD-R、DVD+R。可擦写DVD光盘有DVD-RW、DVD-RAM和DVD+RW,DVD-RW可重写1 000次以上,DVD-RAM是可随机存取的光盘,可重写10 000次以上。DVD+R和DVD+RW分别基于飞利浦公司提供的一次写和可擦写光盘技术格式,未得到DVD论坛的认可。复合DVD光盘不是新的格式和标准,而是已有不同DVD格式的组,从而与不同DVD格式最大程度地兼容。

DVD光盘根据其物理结构可分为单面单层、单面双层、双面单层和双面双层四种规格,其中每种规格根据盘片直径又可分为120 mm和80 mm两种,最常用的是120 mm。只读DVD光盘具有上述四种规格,其余均为单面单层。120 mm单面单层只读DVD光盘容量为4.7 GB(称为DVD 5),单面双层容量为8.5 GB(称为DVD 9),双面单层容量为9.4 GB(称为DVD 10),双面双层容量为17 GB(称为DVD 18)。由于工艺过于复杂,一般不制作DVD 18光盘。

硬件结构

DVD光学头是DVD光盘机的核心部件,由半导体激光器、物镜、光电探测器和执行器等相关光学及机电元件组成,实现信息的读出、写入及擦除。

DVD伺服系统实现光学头读写光斑在光盘信

息道的聚焦、循迹和扫描功能,包括聚焦伺服系统、循迹伺服系统、寻址伺服系统和主轴伺服系统。聚焦伺服系统实现光斑的轴向定位控制,循迹伺服系统实现光斑的近距离径向定位控制,寻址伺服系统实现光斑的长行程径向定位控制,主轴伺服系统通过控制光盘的转速实现光斑对光盘信道的恒线速或恒角速等的扫描。

DVD解码系统实现信道解码和音视频解码功能。除DVD光学头之外,伺服和解码均由专用芯片实现。

发展趋势

DVD技术在提高数据存储密度和数据传输速率两个方向发展。蓝光光盘技术可以将单面单层盘片容量提高到25~30 GB,多波长多阶光盘存储、全息存储和宽光束存储等新技术均可显著提高存储密度和数据传输速率。DVD技术与网络技术相结合,实现信息家电与计算机的融合是DVD应用技术发展的一大趋势。

参考文献

徐端颐. 光盘存储系统设计原理. 北京:国防工业出版社,2000
(潘龙法)

shuzihuayi

数字化仪(digitizer) 将图形信息转换成相应的二进制码并输入到计算机的设备。

数字化仪由绘图板和游标器或画笔组成。绘图板是一个平整的平面,板上有检测 x, y 坐标的网格传感器,以确定平面每一点的坐标。常用的是电磁感应式传感器,也有用光电式、电容式、电阻式、磁致伸缩式传感器的。通常绘图板中央部分用于图形输入,两旁有各种常用图形符号供选用。游标器类似于鼠标器,所不同的是游标器所附的透明塑料板上画有十字形标记定位线。当使用者将十字形标记的交叉点对准板上某一点时,传感器即在屏幕上显示出对应的一点。使用者按下相应的按钮,就可在这一点上进行所需的操作。也可以用游标器对准两旁的某一图形,按下按钮,将该图形送入计算机进行处理。利用数字化仪可以很方便地在计算机屏幕上作图。20世纪70年代以来,数字化仪已在计算机辅助设计和计算机辅助工程领域中得到广泛应用,是交互式绘图常用的输入设备。

数字化仪的绘图板面积大的可达1 118 mm × 1 524 mm(44 in × 60 in),分辨率可做到0.01 mm。

20世纪90年代以来,小型数字化仪得到很好

的发展,它是手写汉字识别系统的输入设备。

(林兼)

shuzi huayin de yasuo bianma

数字语音的压缩编码 (compression and coding of digital voice)

对数字语音进行数据压缩和编码的方法与技术。语音是人们说话时肺部的空气沿着声道通过声门发出时产生的,其信号带宽比较窄。例如电话语音信号的带宽只有 300 ~ 3 400 Hz,取样频率一般为 8 kHz,量化精度为 8 位,每秒钟的数据量(码率)是 8 kB,1 小时的数据量大约是 28 MB。为了降低存储成本和提高通信效率(降低传输带宽),对数字语音进行数据压缩是十分必要的。

由于声音信号中包含有大量的冗余信息,再加上还可以利用人的听觉感知特性和语音信号的生成机理,因此,大幅度压缩数字语音的数据是完全可能的。目前,数字语音的数据压缩方法很多,从原理上大致可以分成如下 3 类。

(1) 波形编码 这是一种基于感觉模型的压缩方法,例如 PCM(脉冲编码调制)、ADPCM(自适应差分脉冲编码调制)、子带编码等。它们的码率虽然比较高(分别为 64 kb/s 和 32 kb/s),但能保证语音的高质量,且算法简单、易实现,在电话中继通信中得到广泛应用,如 CCITT G. 711 和 G. 721。此类方法通用性好,不仅适用于数字语音的压缩,而且对所有波形形式的数字声音都有效。不足之处是很难达到很高的压缩比。

(2) 模型编码 又称参数编码或源编码。这是一种基于语音生成模型的压缩方法,它使用“声源-滤波器”模型来模拟人的发声过程。压缩编码时,使用线性预测方法从原始的话音波形信号中提取语音的生成参数,并把这些参数作为原始语音压缩编码的结果,因此码率很低。解码时必须使用与编码完全相同的模型,所以可应用于保密通信,例如声码器。此类方法的优点是压缩比很高,但质量还不很理想。

(3) 混合编码 又称“合成-分析法”(AbS),这是上述两种方法的结合。它利用原始的话音波形信号提取“声源-滤波器”模型中的声道参数与激励信号,并要求使用这些激励信号所产生的波形应尽可能与原始语音的波形一致。采用此类压缩编码方法后,数字语音的码率一般在 4.8 ~ 16 kb/s,达到了较高的压缩比,同时又能保证较好的语音质量。其中使用最多的是码激励线性预测法(CELP),目前移动

通信和 IP 电话中话音信号的压缩编码大多采用这种方法。

图 1 是上述 3 类数字语音压缩编码方法的比较。

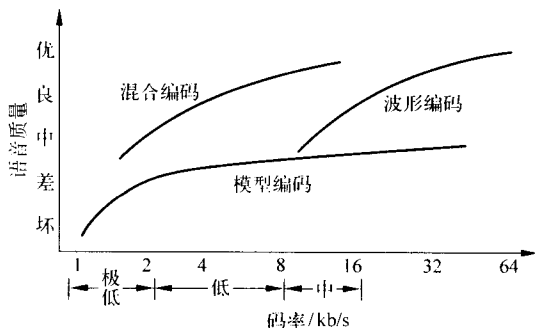


图 1 数字语音压缩编码方法的比较

参考文献

Kondoz A. M. Digital Speech Coding for Low Bit Rate Communication Systems. John Wiley & Sons, 1999

(张福炎)

shuzi jicheng dianlu

数字集成电路 (digital integrated circuit)

基于布尔代数(逻辑代数、开关代数)的公式及规则,能对二进制数进行布尔运算的集成电路。由于数字集成电路(数字 IC)的输入和输出满足一定的逻辑关系,又称数字逻辑电路。它主要是指由门电路和记忆电路组成,且能实现一定逻辑功能的集成电路,这些逻辑功能包括数字逻辑运算、存储、传输及转换等。数字 IC 是数字电子计算机和所有数字电子系统的硬件基础。

数字 IC 按其内部有源器件的不同可分为两类:一类为双极型晶体管集成电路;另一类为绝缘栅场效应晶体管集成电路(或称 MOS 型电路)。在工艺上,数字 IC 已从当初的电阻-晶体管逻辑(RTL)、二极管-晶体管逻辑(DTL)发展到了晶体管-晶体管逻辑(TTL)和射极耦合逻辑(ECL),同时出现了 PMOS、NMOS、CMOS 和 GaAsMOS 等 MOS 型数字 IC。表 1 为主要数字 IC 类型及其电性能比较。

数字 IC 主要由门电路组成,因之可按每个芯片上集成的等效门电路个数或元器件个数来表征该芯片的集成度。通常按集成度将数字 IC 分为 6 类,即小规模集成电路(SSI)、中规模集成电路(MSI)、大规模集成电路(LSI)、超大规模集成电路(VLSI)、特

大规模集成电路 (ULSI) 和巨大规模集成电路 (GLSI), 如表 2 所示。作为比较, 表中列出了晶体管和分立元件的发展年代。

表 1 主要数字 IC 类型及电路性能比较

特性 电路类型	电路结构	功耗	速度 特性	集成特性
RTL	双极型	高	低	分立
DTL	双极型	高	低	分立、SSI
TTL	双极型	中	中	SSI、MSI
ECL	双极型	高	高	SSI、MSI、LSI
PMOS	MOSFET	中	低	MSI、LSI
NMOS	MOSFET	中	中	MSI、LSI、VLSI
CMOS	MOSFET	低	中	SSI、MSI、LSI、 VLSI、更高
GaAs	MESFET	高	高	SSI、MSI、LSI

表 2 集成电路的集成度分类

年份	1948	1961	1966	1971	1980	1990	1998	2000
类别	晶体管	分立元件	SSI	MSI	LSI	VLSI	ULSI	GLSI
芯片所含 等效门电 路个数	1	1	小于 10	$10 \sim 10^2$	$10^2 \sim 10^4$	$10^4 \sim 10^6$	$10^6 \sim 10^8$	大于 10^8
芯片所含 元器件 个数			小于 10^2	$10^2 \sim 10^3$	$10^3 \sim 10^5$	$10^5 \sim 10^7$	$10^7 \sim 10^9$	大于 10^9
代表产品	二极管 三极管		门电路 触发器	计数器 加法器	8 位微 处理器	16 位、 32 位 微处理器	图像处理, SOC 文档微处理器	

门电路是组合电路中的基本电路, 它按照输入端条件产生输出。基本门电路有“与”门、“或”门、“非”门、“与非”门、“或非”门、“与或非”门、“异或”门、“同或”门等。表 3 中以 A、B 双输入和 Q 输出为例给出基本门电路的逻辑表达式、电路符号表示和真值表。

图 1 给出了四种典型电路图。以图 1(b) 为例, 当 A、B 同时为高电平时, T_1 截止, T_2 导通, 导致 T_3 , T_4 截止, T_5 导通, 输出为低电平; 当 A、B 中有一个为低电平时, T_1 导通, T_2 截止, 导致 T_3 , T_4 导通, T_5 截止, 输出为高电平。

具有复杂功能的组合电路可由上述的基本门电路构成。由于中、大规模集成电路技术的进

按照芯片的设计方法, 数字 IC 可分为通用集成电路(如市售的小、中、大规模集成电路产品)、可编程逻辑器件(如 PROM、EPROM、FPLA、PAL 等)、半定制集成电路(如门阵列、标准单元等构成的集成电路)和全定制集成电路。

按其输出的生成方法, 数字 IC 可分为组合(逻辑)电路和时序(逻辑)电路两种。组合电路的输出只取决于当前输入的组合, 而不受过去输入的影响; 时序电路的输出则取决于当前输入和过去输入所引起的当前状态。

数字 IC 分别以高、低电平表示逻辑 1 和 0。若 IC 用高电平表示逻辑 1, 低电平表示逻辑 0, 称为正逻辑电路; 反之, 高电平对应逻辑 0, 低电平对应逻辑 1 的电路称为负逻辑电路。这是在设计数字 IC 时约定的。

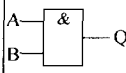
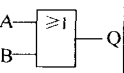
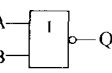
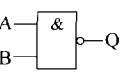
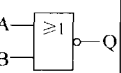
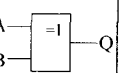
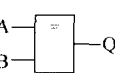
步和成本的降低, 可以把由许多个基本门电路构成的组合电路作为基本 IC 使用。这类组合电路有译码器、编码器、数据选择器、数值比较器、模拟开关、算术逻辑部件、奇偶产生器、奇偶检验器、只读存储器和可编程组合逻辑器件(如 PLA, PAL, FPLA)等。

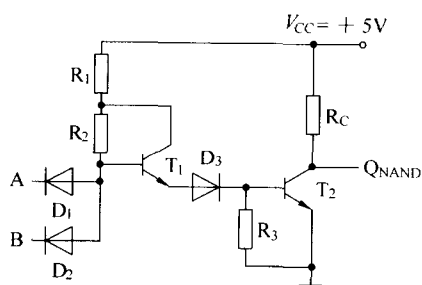
触发器是时序电路中构成存储电路最常用的存储元件, 它的输出不仅依赖于当前输入端的状态, 而且依赖于当前该电路的内部状态, 而该内部状态又依赖于前一时刻电路输入端的状态和前一时刻的内部状态。基本的触发器电路有 R-S 触发器、T 触发器、J-K 触发器和 D 触发器。图 2 是 R-S 触发器电路图, 它由两个“与非”门或者两个“或非”门电路加

反馈构成。门的输入端分别为复位 R 、置位 S ，输出端为 Q 和 \bar{Q} 。当 $S=1, R=0$ 时, $Q=1$ (置位状态); 当 $S=0, R=1$ 时, $\bar{Q}=1$ (复位状态); $R=S=0$ 时, 保

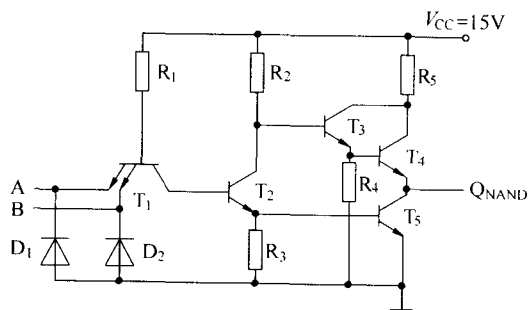
持原状态不变; $R=S=1$ 时, Q, \bar{Q} 状态不定。图 3 为上述 4 种触发器的电路符号和工作特性。

表 3 基本门电路的逻辑表达式、电路符号表示和真值表

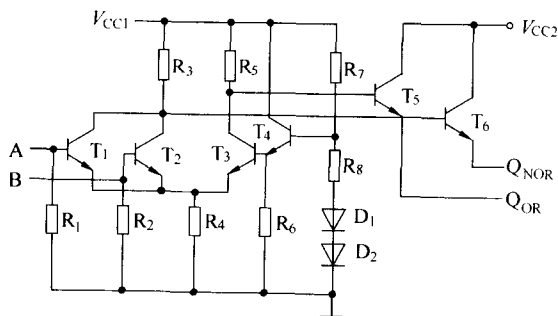
逻辑门名称 项 目	“与” 门 AND	“或” 门 OR	“非” 门 NOT	“与非” 门 NAND	“或非” 门 NOR	“异或” 门 EOR	“同或” 门 ENOR																																																																																																
逻辑表达式	$Q=AB$	$Q=A+B$	$Q=\bar{A}$	$Q=\overline{AB}$	$Q=\overline{A+B}$	$Q=\bar{A}B+A\bar{B}$	$Q=AB+\bar{A}\bar{B}$																																																																																																
电路符号表示																																																																																																							
真值表	<table><tr><td>A</td><td>B</td><td>Q</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Q	0	0	0	0	1	0	1	0	0	1	1	1	<table><tr><td>A</td><td>B</td><td>Q</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Q	0	0	0	0	1	1	1	0	1	1	1	1	<table><tr><td>A</td><td>Q</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	Q	0	1	1	0	<table><tr><td>A</td><td>B</td><td>Q</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Q	0	0	1	0	1	1	1	0	1	1	1	0	<table><tr><td>A</td><td>B</td><td>Q</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Q	0	0	1	0	1	0	1	0	0	1	1	0	<table><tr><td>A</td><td>B</td><td>Q</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Q	0	0	0	0	1	1	1	0	1	1	1	0	<table><tr><td>A</td><td>B</td><td>Q</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Q	0	0	1	0	1	1	1	0	0	1	1	1
A	B	Q																																																																																																					
0	0	0																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					
A	B	Q																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	1																																																																																																					
A	Q																																																																																																						
0	1																																																																																																						
1	0																																																																																																						
A	B	Q																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
A	B	Q																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	0																																																																																																					
A	B	Q																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
A	B	Q																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					



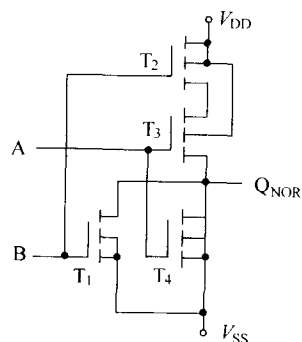
(a) DTL “与非”门电路



(b) TTL “与非”门电路 (74H 系列)



(c) ECL “或非”、“或”电路



(d) CMOS “或非”门电路

图 1 基本逻辑门电路图

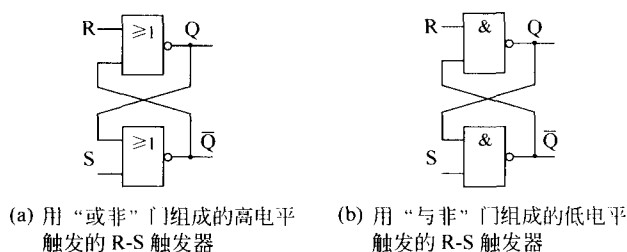


图2 基本 R-S 触发器电路图

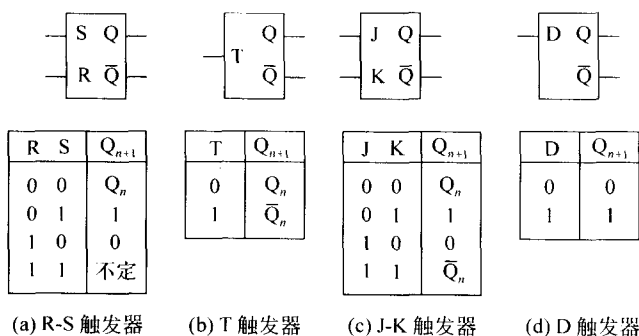


图3 基本触发器的电路符号及其工作特性

具有复杂功能的集成时序电路主要有寄存器、移位寄存器、计数器、可读写存储器(RAM)和可编程时序逻辑器件(如 RPLA、GAL、FPGA)等。寄存器和移位寄存器都是计算机中用来暂时寄存数码的存储元件,它们都具有接收、存储和传送数码的功能。移位寄存器还具有移位功能,可以单向移位或双向移位。按寄存器和移位寄存器的工作方式和输出稳定状态可分为静态和动态两种;若按输入、输出形式可分为串入-并出、串入-串出、并入-串出和并入-并出四种。计数器是计算机中广泛应用的一种可以记录脉冲个数的电路。计数器按计数制可分为二进制计数器、十进制计数器、二十进制计数器等;按其计数功能可分为加计数器、减计数器和可逆计数器;按预置和清除方式则可有并行预置、直接预置、同步清除和异步清除。计数器还可按工作方式分类,有同步计数器和异步计数器。

由于数字 IC 容差大、体积小、重量轻,因而得到了迅速发展,其应用越来越广泛,内容越来越丰富,技术越来越成熟。在许多情况下,完成同样功能,数字 IC 与模拟 IC 相比,速度更快、精度更高、工作更稳定、抗干扰能力强、实现更容易、操作更简便,此外在信息的处理、储存、故障检测与校正等方面更加方便。数字 IC 正朝着更加高速、低耗、低电源电压和

高集成度方向发展。

参考文献

1. Victor P Nelson, Troy H Nagle, Bill D Carroll, David J Irwin. Digital Logic Circuit Analysis and Design. Prentice Hall, 1995. 北京:清华大学出版社, 1997(影印版)
2. 尤忠琪,贾立新. 数字集成电路教程. 北京:科学出版社, 2001

(时万春)

shuzi jihe chuli

数字几何处理(digital geometry processing)

以数字化的几何数据及相关的附着物理属性为对象,部分借鉴传统信号处理方法,按特定目的对数据实施去噪、压缩、转换、增强、检测和分析等操作。广义的数字几何处理流程包含:①创建和获取;②存储与传输;③认证;④编辑与动画;⑤仿真计算;⑥加工制造等阶段。

数字几何数据是一种不同于图像、音频和视频的新数据媒体。随着三维信息获取技术的不断成熟,三维几何模型在影视娱乐、设计与制造、电子商务和模拟仿真等领域得到了广泛的应用,从而推动了数字几何处理技术的产生和发展。

在处理的不同阶段,数字几何数据采用了多种

表示方法,根据数据的表述形式,可概要区分为连续和离散两种数据类型。离散数据,包括体素、多边形网格和点云数据,是通过一定方式对物体的几何信息进行采样获得。该类数据往往出现在整个处理的早期,即数据获取阶段。由于多边形网格数据的结构简单,利于绘制和表示复杂形体,适于与其他数据类型转换,因此其应用贯穿了整个数字几何处理过程,成为主要研究内容(详见**三维网格处理**)。连续数据采用显式或者隐式的连续数学解析表示,其优点在于表示紧凑,利于给出任意精度的采样。再者,如**参数曲面**等显式表示,可快速赋值,适于直接的曲面形状交互和控制。而以径向基函数曲面为代表的**隐函数曲面**则便于求交计算,具有良好的插值和外推能力,可用于填补获取数据的缺失部分,避免曲面的局部重叠。目前,研究者还发展出了多种过程式和混合式的表示处理方法,如结合**参数曲面**和多边形网格优点的细分方法,结合空间八叉树**剖分曲面**生成技术来提高赋值和查询速度等。

几何数据上附着的物理属性主要指物体的外观描述,在计算机中常被描述为颜色、纹理、光泽和透明度。由于真实世界的物体表面细节极其复杂,人们研究了多种经验和物理模型,如双向反射函数、**纹理映射**、凹凸和位移纹理,以及双向反射纹理等。在数字几何处理中,这些附着属性与三维几何位置信息一样,一般关联于对应顶点,因而可通过增加顶点信息的维数而纳入原有的几何处理框架。

不同于其他3种数字媒体的处理,由于其数据特性,使得数字几何处理对目前的计算框架提出了重大挑战。主要的难点有:①获取的离散数据往往是非均匀采样,噪声产生原因复杂,存在数据残缺现象;②数据的几何和拓扑依赖性强,不利于使用原有的信号处理框架;③数据量变化剧烈,海量数据在未经处理时往往超过目前计算机的处理和显示能力,需考虑合理分片和简化技术。

参考文献

1. Schröder P and Sweldens W. Digital Geometry Processing. ACM SIGGRAPH 2001 Course, August 2001, Los Angeles, California

2. 周昆. 数字几何处理:理论和应用. 浙江大学,博士论文,2002
(鲍虎军 张宏鑫)

shuzi jisuanji

数字计算机(digital computer) 对用离散符

号表示的信息在程序控制下自动进行处理的电子装置,其全名是电子数字计算机。由于这类**电子计算机**已经普遍应用于人类社会的方方面面,通常人们提到的计算机往往就是指数字计算机。

数字计算机作为一个信息处理系统,包括**硬件**和**软件**两大部分。硬件是组成计算机的电子器件及其线路、部件和外围设备等的统称;软件则是使计算机能够自动运作以完成信息处理任务的各种程序及其文档的统称。下面扼要介绍计算机核心结构的组成和工作原理。

计算机硬件系统的组成

计算机的硬件系统主要由中央处理器、主存储器和外围设备组成。当然,支持和保障计算机得以正常运行的机箱、电源、通风散热设施等也是必不可少的,因它们不涉及计算机的工作原理,以下不再提起。下面将分别介绍各组成部分的主要功能。

中央处理器(CPU) 主要由控制器、运算器、内部寄存器组和处理机总线等部分组成(参见**中央处理器**)。控制器的主要功能是自动从主存储器读取指令,予以解释和执行,并控制与内部寄存器组、主存储器或外围设备交换信息和数据;运算器的主要功能是按照指令的要求完成相应的算术或逻辑运算;处理机总线的功能是将CPU内部的各部件连接起来。

主存储器(MM) 简称主存,亦称内存。它是可随机存取的操作存储器,所以也称**随机存储器(RAM)**(参见**主存储器**)。它存储计算机运行时随时所要用到的程序和数据。现代计算机采用快速大容量的**半导体存储器**。

外围设备 输入输出设备和外存储器的统称。计算机运行时所需要的程序和数据都要由输入设备输入到内存中。计算机在进行信息处理的过程中所产生的最终结果和某些中间结果要通过输出设备送出。受内存容量的限制,加以半导体RAM中的信息在断电时会全部消失,所以需要有大容量的、能长期保存数据的外存储器。常用的输入装置是**键盘**和**显示器**。显示器同时也是输出显示装置,是操作员和用户与计算机相互联系的窗口。其他常用的输出装置有针式打印机、激光印刷机、喷墨印刷机、**扫描仪**、**绘图仪**、**数码摄像机**等。常用的外存储器是**磁盘存储器**和**磁带存储器**。磁盘又有**软磁盘**和**硬磁盘**之分。软磁盘和磁带因便于装卸,又可脱机保存数据,必要时也可兼作输入媒体用。光盘能够不受电磁干扰,并能够长期保存大量信息。

计算机中信息的表示

计算机采用二进制编码方式表示数、字符、指令和其他控制信息,这不但比十进制表示简单,而且便于用最简单的开关电路实现二值逻辑运算。计算机在存储、传送或处理数据时,作为一个单元的一组二进制代码称为**字**,一个字中的二进制位的位数称为**字长**。常用的字长有8位、16位、32位或64位。字长为8位的代码称为**字节**,它是计算机中表示符号或字符的基本编码单位。

数的表示(参见**数制**) 人们早已习惯用十进制来表示数,其要点是“逢十进一”,即用0,1,⋯,9十个值中的任一个值来表示一位十进制数,当一个数大于9时,就需用多位来表示。在十进制记数系统中,第一位为个位,其权值 $R^0 = 10^0 = 1$;第二位为十位,其权值 $R^1 = 10^1 = 10$;⋯⋯即每进一位,其权值增加十倍。多位数的值等于各位数的值之和。十进制数 N_{10} 可表示为:

$$N_{10} = \sum_{i=1}^n r_i 10^{i-1}$$

式中 n 为位数;

$$r_i = 0, 1, \dots, 9。$$

例如, $1\ 036 = 1 \times 10^3 + 0 \times 10^2 + 3 \times 10^1 + 6 \times 10^0 = 1\ 000 + 0 + 30 + 6。$

相应地,二进制数 N_2 可表示为:

$$N_2 = \sum_{i=1}^n b_i 2^{i-1}$$

式中 $b_i = 0, 1。$

上述十进制数 1 036 可用二进制表示为:

$$\begin{aligned} N_2 &= 10000001100 \\ &= 1 \times 2^{10} + 0 \times 2^9 + 0 \times 2^8 + 0 \times 2^7 \\ &\quad + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 \\ &\quad + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 \\ &= 1\ 024 + 8 + 4 = 1\ 036 \end{aligned}$$

对于负数,一般用数的最高位作为符号位。符号位为0表示正数,为1表示负数。符号位以外的数位若为数的绝对值,这样的数码称为**原码**。除了用原码表示数外,还可用**补码**或**反码**表示数(参见**数制**)。

在计算机内直接表示一个整数或小数方法称为**定点表示法**,因为小数点总是指定在某一固定位置而得名。另一种表示数的方法是**浮点表示法**。这是把一个字分为两段:前一段表示带符号的阶码 p ,即表示 $2^{\pm p}$, p 为整数;后一段表示带符号的定点小

数 m ,称为尾数。用浮点表示法表示的数为: $N_2 = \pm 2^{\pm p} m$,符号位在最前面。显然浮点表示法所表示的数的范围比定点表示法的大得多。

在计算机中,有时用八进制或十六进制表示数,可分别用3位或4位二进制数来表示。一个十进制数也可用4位二进制数表示,这种数称为**二-十进制数**。用二进制数表示十进制数有多种编码形式,包括8421码、余3码、格雷码等。

字符的表示 国际通用的字符编码标准是**ASCII码**。用一个字节中的低七位编码表示 $2^7 = 128$ 个字符,最高位用于奇偶检验或置为0。字符包括十进制数码、大、小写英文字母、标点符号、常用算符、各种控制符和特殊标志等。同样也可以用二进制数编码表示汉字。由于一个字节的8位编码最多只能表示 $2^8 = 256$ 个不同的字符,远远少于不同汉字的数量。所以,至少必须用两个字节来表示不同汉字的编码。按标准规定,令其中每个字节的最高位为“1”,以示与ASCII码的区别(参见**字符集**)。

指令的表示 指令是指示计算机操作的命令。通常用一个以上字节的编码来表示一条指令。指令主要由操作码、地址码等部分组成。它们所占用的位数,决定操作种类的数量和最大可访问的存储空间。例如,7位操作码可以表示 $2^7 = 128$ 种不同的操作;10位地址码的寻址空间为 $2^{10} = 1\ 024$,称为1 K。16位地址码的寻址空间为 2^{16} ,即64 K。此外还要设置若干位来表示指令的特征和寻址方式等。

数字和字符的编码表示统称为**数据**,它们是指令的操作对象。指令和数据同样用二进制编码形式表示,这是计算机得以在程序控制下完全自动进行数据处理的关键。

逻辑线路

二进制数位的两个值0或1,正好与命题逻辑的“真”和“假”一一对应。于是,二进制的运算可以很自然地转化为**逻辑运算**。最基本的逻辑运算为:①“非”,记为 \bar{A} 或 NOT A 。即,若 $A = 1$,则 $\bar{A} = 0$;反之亦然。②“或”,记为 $A \vee B \vee \dots$ 或 A OR B OR \dots ,表示命题 A, B, \dots 中只要有一个或一个以上命题为真,其结果即为真。相应的布尔表达式为 $Z = A + B + \dots$ 。③“与”,记为 $A \wedge B \wedge \dots$ 或 A AND B AND \dots ,表示当且仅当全部命题 A, B, \dots 皆为真,其结果方为真。相应的布尔表达式为 $Z = AB \dots$ 或 $Z = A \times B \times \dots$ 。由于二进制数只有0和1两个值,所以,可用电子开关电路,或称“门”电路来实现相应的运算。与上述3种基本的逻辑运算相对应的基本逻辑

线路是“非门”、“或门”和“与门”。它们可以组成计算机的各种不同功能的逻辑线路。例如,用两个“或非门”或两个“与非门”可以组成一个触发器,可作为一个二进制码的存储单元;多个触发器并列且与相应的输入、输出控制门连接,可组成能存储一个字的寄存器。还可以用基本逻辑电路组成加法器、译码器、编码器、时序信号发生器、分频器、移位寄存器等。前述计算机的各主要部件,如 CPU、MM、外围接口等极其复杂的线路,也都是由简单的基本逻辑电路加上一些辅助电路所组成。

存储程序概念

计算机 ENIAC 开创了电子计算机发展的新纪元。但是,控制其运算的程序却是在类似于模拟计算机的排题板上,用外接线接插编排的。早在 ENIAC 的研制过程中,其主设计师 J. P. Eckert 和 J. W. Mauchly 就有了用超声波延迟线作为二进制代码的串行存储器来动态保存指令的想法,并在美籍匈牙利数学家冯·诺依曼的参加下,着手制定了重新研制一台能把程序和数据一同存储在操作存储器中,从而实现在存储程序控制下自动计算的计算机 EDVAC 的计划。冯·诺依曼在由他执笔的“关于 EDVAC 设计方案的初步报告”(1945 年 6 月 30 日)中,首次明确地阐述了存储程序概念的基本设计思想。冯·诺依曼等于 1946 年发表的题为“电子计算装置逻辑设计的初步讨论”的论文中,进一步系统地深入阐述了以存储程序概念为指导的计算机逻辑设计思想,勾画出一个完整的计算机体系结构,至

今仍不失其开创性的指导意义。后来人们称这种体系结构为冯·诺依曼结构;称采取这种体系结构的计算机为冯·诺依曼计算机。EDVAC 虽然于 1945 年开始研制,但由于种种原因,到 1951 年才正式投入运行。

冯·诺依曼结构的核心是存储程序概念。其要点为:①指令与数据均储存在主存储器中;②在 CPU 的控制器中,设置一个程序计数器 PC,其初始值置为程序的第一条指令所在存储单元的地址。计算机一旦被启动,控制器就会自动按 PC 的当前值所指,从主存储器读出指令,予以解释、执行。同时,PC 自动加 1,指向顺序执行的下一条指令。执行每条指令所涉及的操作数,则按指令中的操作数地址所指,从主存储器读取或向主存储器写入。以上要点如图 1 所示。要执行的程序和有关数据一同存放在内存(即主存储器)的不同区间。开始执行程序时,其首地址放在 CPU 中特设的程序计数器(PC)中。按 PC 中的首地址读出第一条指令,传送到指令寄存器(①,②),同时 PC 自动加 1,准备好读取顺序存放的下一条指令。执行指令所涉及的数据则按指令寄存器中的有效地址码所指的存储单元,进行读出或写入(③,④)。值得一提的是,某些指令(例如转移指令等)中的地址可以是另一条指令或另一段程序(例如子程序、中断服务程序、另一个分时运行的程序等)的指令所在存储单元的地址,这就增加了自动调度程序的灵活性。现代计算机正是这样得以实现在程序控制下全自动地进行信息处理。

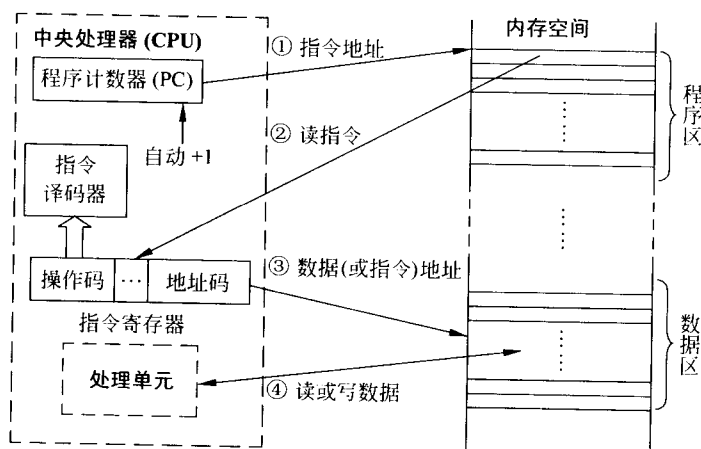


图 1 存储程序控制的原理

计算机的程序设计语言

由图 1 可见,在所执行的程序代码已经存放在

内存的前提下,计算机的工作过程就是从读出第一条指令开始,顺序读出、解释和执行每一条指令的过

程。这些用二进制编码表达的指令构成了计算机能直接“理解”并严格执行的代码语言,称为**机器语言**。这种语言与人类的自然语言相差很远,直接用它来编写程序是一项十分费时、难懂且很容易出错的工作。因此,早在计算机发展初期的1953年,就开始采用所谓“助记符”的符号来书写指令,从而产生了**汇编语言**。用汇编语言编写的程序,先要通过**汇编程序**转换为代码程序(亦称目标代码),再由计算机来运行。为了进一步从根本上克服用机器语言编程所带来的种种弊端,Backus在1957年首先提出第一个高级程序设计语言FORTRAN,并设计了相应的**编译程序**(或称**编译器**)。这种编译程序使人们得以摆脱计算机指令系统和机器语言的约束,可以用易写和易懂的形式语言来编写程序。此后,各种**高级程序语言**相继出现,包括ALGOL-60, BASIC, COBOL, PASCAL, C, C++, LISP, Ada, PROLOG, Java等。

参考文献

1. Carl Hamacher等. 计算机组成(第5版. 影印版). 北京: 机械工业出版社, McGraw Hill Education, 2002
2. 葛本修等. 计算机系统组成及工作原理(计算机等级考试教程: 四级). 北京: 机械工业出版社, 2000
3. 王爱英主编. 计算机组成与结构(第3版). 北京: 清华大学出版社, 2001 (童颖)

shuzi kexie guangpan

数字可写光盘(compact disc-recordable, CD-R) 采用可写一次记录材料作为存储介质的无盘盒结构的数字光盘。CD-R标准(橙皮书)是由Philips公司于1990年制定的,后成为工业界广泛认可的标准。CD-R只允许写一次,写完以后,记录在CD-R盘上的信息无法被改写,但可以在CD-ROM驱动器、DVD-ROM驱动器、CD-R驱动器和CD-RW(数字可重写光盘)驱动器上被反复地读取多次。CD-R盘与CD-ROM盘相比有许多共同之处,它们的主要差别在于CD-R盘上用一层有机染料作为记录层,反射层用金或银,而不是CD-ROM中的铝。当写入激光束聚焦到记录层上时,染料被加热后烧溶,形成一系列代表信息的凹坑。这些凹坑与CD-ROM盘上的凹坑类似,但CD-ROM盘上的凹坑是用金属压模压出的。

CD-R盘片的结构与原理

CD-R光盘的外形尺寸与CD-ROM光盘完全一样,其直径为120 mm,厚度约为1.2 mm。与CD-ROM光盘三层结构所不同的是,CD-R光盘采用了四层结构形式,底层为聚碳酸酯透明塑料注塑成形的衬盘,在塑料衬底上镀有一层很薄的有机染料记录层,并使用抗腐蚀的金膜作反射层,可以使CD-R光盘达到70%的反射率,反射层外为涂漆保护层。有些CD-R光盘在漆保护层之上还用吸墨材料涂有第五层印刷层,用户可用喷墨打印机直接在CD-R光盘背面打印商标、图案或文字说明,也可用软笔进行标注。

CD-R光盘的塑料衬盘上预先刻有宽度为 $0.6\mu\text{m}$ 、深度为 $0.1\mu\text{m}$ 的螺旋形预刻凹槽,预刻槽由CD-R光盘的内圈以螺旋状展开,一直延伸到盘的最外圈结束。CD-R和CD-RW刻录机将通过对预刻槽的跟踪和聚焦,来引导激光束对CD-R光盘进行刻录或读取。

CD-R光盘记录层中的有机染料对波长为780 nm的光波具有很强的吸收作用,当波长为780 nm的半导体激光器输出的聚焦激光束照射在有机染料记录层时,光点处的有机染料将吸收很大的能量,并瞬间将吸收的能量转化成热量,在受聚焦激光束照射的微区产生250℃到400℃的高温,使有机染料溶解气化,在塑料衬盘和反射层之间形成一微孔,使微孔区域的反射层完全暴露出来,从而使微孔处具有很强的光反射作用。

刻录CD-R光盘时,CD-R光盘以恒定线速度旋转,功率可调的激光束沿着螺旋形预刻槽烧出一系列相间的不同长度尺寸的微孔,以实现记录数据的正确写入。由于有机染料记录层的微孔处具有强的光反射特性,而未溶解气化的有机染料反射率很低,因此CD-R光盘上的微孔与未溶解的有机染料分别具有与CD-ROM光盘上的平台和凹坑相同的光学反射特性,可以利用CD-ROM驱动器和DVD-ROM驱动器进行数据读取。

CD-R光盘的种类

CD-R空白光盘主要有绿盘、金盘和蓝盘三种类型。这三种光盘的主要区别在于,它们分别使用了花菁、酞菁和金属化偶氮(AZO)化合物三种不同颜色的有机染料,从而使CD-R光盘呈现出绿、金、蓝三种不同的颜色。从数据记录和读取的原理来看,不同颜色的CD-R光盘都具有相同的功能。

绿盘 绿盘是最早开发生产出的CD-R光盘,采用了花菁染料。由于花菁染料的颜色为青蓝色,因

此与 24K 金反射层的金色混合之后,就会使 CD-R 光盘的记录面呈现绿色。由于 CD-R 标准(橙皮书)是基于花菁染料的记录灵敏度、记录阈值和反射率等记录特性制定出的,而且所有的 CD-R 或 CD-RW 刻录机均按照橙皮书规格进行设计生产,因此,绿盘对各种品牌和型号的 CD-R 或 CD-RW 刻录机的兼容性较强。绿盘使用的花菁染料的主要问题是绿盘对强光过于敏感,为了降低绿盘对强光的敏感性,一些 CD-R 绿盘生产厂家在花菁染料中加入了不易感光材料,结果使花菁染料的颜色变淡,从而使这种绿盘的颜色与金盘接近,因此这种 CD-R 光盘也被称为金绿盘。

金盘和白金盘 针对花菁染料对强光敏感的缺点,出现了酞菁染料。酞菁染料本身呈淡黄色,与反射层的金色混合后,使 CD-R 光盘的记录面呈黄金色。因此,使用金反射层的酞菁染料 CD-R 光盘被称为金盘。为了降低金盘的生产成本,目前许多 CD-R 盘生产厂使用银反射层,酞菁染料的淡黄色与反射层的银色混合后,使记录面呈白金色,故这种 CD-R 盘被称为白金盘。

与花菁染料相比,酞菁染料具有较高的稳定性,对室内和室外强光均不敏感。从苛刻实验条件下获得的实验结果可以推断出,CD-R 金盘的使用寿命超过 100 年,这说明 CD-R 金盘适于可靠地长期保存数据。

金盘对 CD-R 或 CD-RW 刻录机的写入激光功率要求较高,酞菁染料通常推荐的写入激光功率为 $(6.5 \pm 0.5) \text{ mW}$;而绿盘对写入激光功率的要求较低,花菁染料的写入激光功率为 $(5.5 \pm 1.0) \text{ mW}$ 。因此,绿盘较低的写入功率和较宽的功率范围可降低对 CD-R 或 CD-RW 刻录机写入激光功率的要求,大大提高了绿盘与 CD-R 或 CD-RW 刻录机的兼容性。

蓝盘 为了降低 CD-R 绿盘和金盘的成本,又出现一种金属化的 AZO 有机染料,并使用成本较低的银作反射层材料。AZO 本身为深蓝色,因此与反射层的银白色混合后,使 CD-R 光盘的记录面呈蓝色,因此使用 AZO 染料的 CD-R 光盘就被称为蓝盘。CD-R 蓝盘除了价格便宜之外,也具有可长期保存数据的优点。实验结果表明,蓝盘也具有 100 年以上的使用寿命。

CD-R 光盘只能写一次的缺点从另一方面来看却是一种优点。由于 CD-R 光盘无法被改写,因此 CD-R 光盘具有极高的安全性。CD-R 光盘为那些需

要长久存储信息,而又不准任何人擦除或更改的用户提供了一种最佳的数据存储解决方案。

CD-R 刻录机

CD-R 刻录机不仅可对 CD-R 空白光盘进行刻写,而且能读取普通的 CD-ROM 和 DVD-ROM 光盘,写完后的 CD-R 光盘可在 CD-ROM 驱动器、DVD-ROM 驱动器、CD-R 驱动器和 CD-RW 驱动器上被读取。

CD-R 刻录机的刻录速度经历了 2 倍速、4 倍速、6 倍速和 8 倍速等发展阶段。由于 CD-RW 刻录机的快速发展(参见数字可重写光盘),单独的 CD-R 刻录机已经逐渐被 CD-RW 刻录机取代。

参考文献

干福熹等. 数字光盘存储技术. 北京: 科学技术出版社, 1998 (黄浩 谢长生)

shuzi kechongxie guangpan

数字可重写光盘 (compact disc-rewritable, CD-RW) 采用相变材料作为存储介质并可以多次重写的无盘盒结构的数字光盘。为了满足众多应用领域对 CD 光盘提出的可多次重复擦写的要求, Philips 公司在 1995 年 4 月制定出了与 CD-ROM 和 CD-R 兼容的相变型可擦写光盘驱动器 CD-E 标准。1996 年 10 月, Philip, Ricoh, Sony, HP 和 Mitsubishi 五家公司在 CD-E 的基础上联合制定了新的可擦写 CD 标准,并将 CD-E 更名为 CD-RW。1996 年底, Philip 和 Ricoh 分别向全世界推出了第一台 CD-RW 可擦写刻录机和第一张 CD-RW 可重写光盘。CD-RW 可擦写刻录机不仅能读取 CD-ROM 光盘,刻录 CD-R 光盘,而且还能对专门的 CD-RW 光盘擦写多次,可担负计算机工作过程中所有信息的暂存和永久保存。

CD-RW 盘片的结构与原理

CD-RW 盘片主要由 6 层组成,分别是盘基、下绝缘层、记录层、上绝缘层、光反射层和保护层。为了达到多次写入的目的,CD-RW 使用了相变技术。它利用一种可逆变结晶材料在结晶与非结晶状态下不同的光特性来记录数据,而这两种状态可以在高、中功率激光的照射下相互转换,所以可以删除并重新写入数据。CD-RW 盘片的记录层即是用这种材料制成。

初次使用的 CD-RW 光盘的记录层处于结晶状态。在进行刻录时,首先按数据中的 0,1 信号的转换来调制刻录激光的照射开关。此时的激光是短

时、高功率的(比刻录 CD-R 的激光更强),被照射的记录层的温度会迅速超过熔点,而在迅速冷却后就变为非结晶状态。此时,结晶的地方可让光线通过到达反射层,没有结晶的地方则很难让光线通过,从而造成了激光反射功率上的差异。在进行数据抹除时,用中等功率的激光对非结晶状态的记录层进行相对长时间的照射,使之慢慢达到结晶的温度,从而转换回原来的结晶状态。为了能让记录层的结晶状态得以长期保持,在它的两边铺上了上绝缘层与下绝缘层来杜绝其与外界环境的联系。另外, CD-RW 盘片在出厂时也已刻好了预刻槽。

CD-RW 盘面一般呈淡灰色,可以反复使用 1000 多次。不过,由于多了好几层物质的阻挡,而且即使是在结晶状态下的透光率也较低,故 CD-RW 盘片的光反射率只有 15% ~ 25%,从而一般低速型号较旧的 CD-ROM 无法读取 CD-RW 盘片。为了解决这个问题,可采用特殊的读取回路,自动调节读取电平,以确保对 CD-RW 盘片反射信号的识别。

在刻录时,可以将 CD-RW 盘片当作 CD-R 盘片来使用,但由于 CD-RW 允许擦写,所以一般都将它当作一个硬盘来用,此时它就好像是一个新的分区,但要达到这一目的,必须先对盘片进行格式化。

CD-RW 刻录机

CD-RW 刻录机可反复读写数据,使在光盘上刻录数据就像在磁盘上存储数据一样容易。为了适应更高密度光盘技术的发展,还出现了将 CD-RW 和 DVD-ROM 技术相结合,可以读取 DVD-ROM 光盘的 CD-RW 刻录机。

CD-RW 刻录机的一个重要指标是缓存的大小,缓存的大小将直接影响到刻录的成功率。据调查统计,至少 80% 的刻录失败都与缓存容量不足有关。CD-RW 刻录机一般使用 4 MB 缓存并向 8 MB 发展。由于在刻录光盘时需要光盘稳定、均匀地旋转,因此对伺服系统的精度、可靠性与寿命都提出了很高的要求,CD-RW 刻录机的伺服系统要普遍好于 CD-ROM 驱动器。

CD-RW 刻录机的另一个重要指标就是速度,一般分为写入速度、重写速度和读取速度三种。这三者一般都以“倍速”为单位来表示,一倍速等于 150 KB/s。主流机型为 40(写) × 12(重写) × 48(读)倍速 CD-RW 刻录机。另外,写入速度与盘片的速度限制密切相关。要想加快写入速度,盘片的旋转速度就要提高,这将使单位面积上盘片所接收到的激光功率减少。高速驱动器需要加大激光的能量输

出,同时,盘片所用介质的灵敏度也要相应提高。CD-RW 盘片都会在其商标一面注明最高速度限制(如 2 ×、4 × 等)。

参考文献

干福熹等. 数字光盘存储技术. 北京: 科学技术出版社, 1998 (黄浩 谢长生)

shuzi qianming

数字签名(digital signatures) 通信双方在网上交换信息时,为防止伪造和欺骗用公钥密码所做的一种身份认证。在传统密码中,由于通信双方用的密钥是一样的,因此,收信方可以伪造、修改密文,发信方也可以抵赖所发过的密文。若产生纠纷,将无法裁决是非(参见密码学)。

由于公钥密码每个用户都有两个密钥,实际上是有两个算法。如用户 A,他有一个算法是公开的加密算法 E_A ,另一个是保密的解密算法 D_A 。若 A 要向 B 送去信息 m ,A 可用自己的保密的解密算法 D_A 对 m 进行加密得 $D_A(m)$,再用 B 的公开的加密算法 E_B 对 $D_A(m)$ 进行加密得

$$C = E_B(D_A(m))$$

B 收到密文 C 后,先用他自己的解密算法 D_B 对 C 进行解密得

$$D_B(C) = D_B(E_B(D_A(m))) = D_A(m)$$

再用 A 的公开算法 E_A 对 $D_A(m)$ 进行解密得

$$E_A(D_A(m)) = m$$

从而得到了明文 m 。

由于 C 只有 A 才能产生,B 无法伪造或修改 C,所以 A 也不能抵赖,这样就达到了签名的目的。不是所有公钥系统都具有数字签名的能力,是 RSA 第一个提供了这样的功能。

参考文献

1. 胡道元. INTRANET 网络技术及应用. 北京: 清华大学出版社, 1998

2. 卢开澄. 计算机密码学. 北京: 清华大学出版社, 1998 (胡道元)

shuzi shexiang tou

数字摄像头(digital PC camera) 采用镜头、光圈等光学系统聚焦成像,通过图像传感器将光学信号转换为模拟图像电信号,内部电路再将模拟图像信号转换为数字图像信号,并通过互连接口传送到计算机的一种数字视频输入设备。

图像传感器是组成数字摄像头的重要组成部分,根据元件不同分为电荷耦合器件(CCD)和互补金属-氧化物-半导体(CMOS)两种类型。两种传感器的差别体现在其图像质量和性能上,其中 CCD 传感器电路比较复杂,但具有较高的图像质量和性能;CMOS 传感器的制造成本和功耗都低于 CCD,通过采用影像光源自动增益补偿技术,自动亮度、白平衡控制技术,色饱和度、对比度、边缘增强以及伽马矫正等先进的影像控制技术,也可以达到与 CCD 传感器相媲美的效果。

数字摄像头与计算机的连接一般通过接口卡、并口和 USB 口这三种方式实现。接口卡方式通过摄像头专用卡来实现,卡上一般都具有摄像头优化或视频捕获功能,在图像画质和视频流的捕获方面具有较大的优势。但由于各厂商的接口卡的设计各不相同,产品之间无法通用。并口方式的优点在于适应性较强,不过数据传输速率较慢。USB 接口方式是主流的连接方式,现有的主机(PC)主板都支持 USB 连接方式。通常数字摄像头的功耗较小,依靠 USB 提供的电源就可以工作,不需要另接电源。

决定数字摄像头质量的主要参数包括:

(1) 分辨率 数字图像分辨率是指整个图像画面垂直和水平方向像素数的乘积。摄像头分辨率是摄像头所拍摄静态或动态图像所能达到的分辨率。例如 640×480 , 352×288 , 320×240 , 176×144 , 160×120 等。一般 PC 用数字摄像头的分辨率可以达到 640×480 ,通过软件插值放大,部分产品最高可达到 704×576 ,使图像、影像表现出丰富的细节,以达到真实的效果。

(2) 像素数 一帧图像中包含的像素的个数。它是由图像传感器上的光敏元件数目所决定的,一个光敏元件就对应一个像素。一般来说,像素越高的产品其图像的品质越好。但另一方面,像素越高,其记录的数据量也必然越大,对于存储设备的要求也越高。

(3) 色彩位数 色彩位数又称位深度,反映摄像头能正确记录颜色种类的多少。色彩位数的值越高,就能更真实地表现亮部及暗部的色彩细节。

(4) 最大帧数 每秒钟捕获的最大视频帧数,又称为视频速度。在一定的数据传输速率下,视频速度和分辨率是直接相关并成反比关系。如果数据传输速率受限,提高分辨率往往会降低视频速度,造成画面跳动的现象。

数字摄像头作为一种计算机辅助配件,一般还

配有专用的控制程序,作为用户直接控制和使用摄像头的工具软件,实现拍照、摄像、设置和管理影像文件等基本功能。(鲁宏伟)

shuzi shipin bianji

数字视频编辑(editing digital video) 采用专门的软件与硬件对数字视频进行各种编辑处理以达到使用要求的过程和技术。

数字视频的编辑处理,不再像模拟视频那样借助磁带录像机进行,而是在大容量的可以随机存取的磁盘存储器上进行。这种摆脱了磁带顺序存取的束缚,使视频节目的制作效率得到了极大的提高,它们被称为非线性编辑系统。非线性编辑系统一般由计算机主机、视(音)频卡、SCSI 硬盘、视频编辑软件和一些控制装置组成。非线性编辑系统的核心软件是数字视频编辑器。编辑时把电视节目素材存入计算机硬盘中,根据需要对不同长短、不同顺序的素材进行剪辑,同时配上字幕、特技和各种动画,再进行配音、配乐,最终制作成高质量的电视节目。

非线性编辑系统实际上是一种视频处理功能特强的多媒体计算机系统,借助于各种软件,它还兼具数字特技机、字幕机、编辑机、调音台的众多功能。以数字视频特技为例,它可以完成多种二维和三维特技控制,例如淡入淡出、变焦、推变等过渡功能,几十种图案划像功能,图像转换功能,图像加边功能,各种图像校正功能等。

数字视频编辑处理的另一个应用是虚拟演播室,它利用计算机生成运动或静止的三维场景,与现场拍摄的视频图像进行实时合成。在现场直播时,虚拟演播室系统综合生成的三维场景可以不断变化和更换,还能根据演播室摄像机运动的位置显示出正确的透视图,使合成的视频图像能取得真实的视觉效果。这样,虚拟演播室技术就使导演摆脱了时间、空间和道具制作等方面的约束,其创造力可以得到充分的发挥。现在,虚拟演播室在电视、电影制作领域中已经发挥着越来越大的作用。

参考文献

张福炎,孙志挥. 大学计算机信息技术教程. 南京:南京大学出版社. 2003 (张福炎)

shuzi shipin huoqu

数字视频获取(Acquisition of Digital Video)

参见图像获取。

shuzi shuju wang

数字数据网 (digital data network, DDN)

一种利用数字信道提供半永久性连接的专用电路,采用时分复用技术,传输以语音、数据、图像、实时信息等数据信号为主的数字传输网络。它向用户提供中、高速率的点到点和点到多点的专用电路,其传输介质一般为光纤、数字微波,用户到用户和端到端传输差错率小于 10^{-6} 。数字数据网(DDN)具有传输质量高、延迟小、通信速率可以根据用户需要自由选择,线路利用率高,用户可以利用 DDN 组建虚拟专网等特点。

DDN 由本地传输系统、复用及交叉连接系统、局间传输及同步系统、网络管理系统等 4 部分组成。

DDN 网络按组建、运营、管理维护的责任和地理区域可分为 3 个网络地域等级:本地网、一级干线网和二级干线网。按功能层次也可分为 3 级:核心层、接入层和用户接入层。

构成 DDN 的干线网或核心层的数字传输系统的信道采用时分多路复用(TDM)。其数字信号分层结构的主要国际标准有美国国家标准局(ANSI)的 DS 系列即 T 系列和欧洲邮政电信管理委员会 CEPT 的 E 系列标准,如表 1 所示。

表 1 DS(T),CEPT(E)系列标准

美国标准 DS(T) 系列		
级别	通道数	速率 Mb/s
DS0	1	0.0064
DS1(T1)	24	1.544
DS1C	48	3.152
DS2	96	6.312
DS3(T3)	672	44.736
DS4	4 032	274.176
欧洲标准 CEPT(E) 系列		
级别	通道数	速率 Mb/s
E0	1	0.0064
E1	30	2.048
E2	120	8.448
E3	480	34.368
E4	1 920	139.264
E5	7 680	565.148

我国的 DDN 向用户提供 2.4 kb/s ~ 2.048 Mb/s 接入速率数据业务服务。

参考文献

1. 海文学,杨谷良.数据通信网基础.北京:人民邮电出版社,1985
2. 陈勤炯.数字数据网——DDN.北京:人民邮电出版社,1994
3. Sidnie Feit. Wide Area High Speed Network. Macmillan Technical Publisher, 1999 (史美林)

shuzi tushuguan

数字图书馆 (digital library) 采用信息技术将图书馆各种载体上的信息以数字化形式进行组织、存储和管理,并可通过计算机网络和各种通信手段为用户提供有效服务的信息集合或信息仓库。数字图书馆也称作数字化图书馆。早在 1975 年 R. W. Christian 在其著作中使用了“电子图书馆”一词,从广义上,将其描述为计算机可处理信息的集合或仓库,但在非图书馆领域则被广泛地用于描述计算机程序或机读数据的集中储藏地。美国密执安大学研究者认为“数字图书馆”可定义为“电子图书馆”。1991 年—1993 年,“电子图书馆”这一术语逐渐向“数字图书馆”转变。1994 年后使用“数字图书馆”的逐渐多起来。在文献中,有的学者在强调这一概念某些侧面特性时也曾使用过“虚拟图书馆”、“无墙图书馆”等术语,但它们与数字图书馆还是有区别的。

美国密执安大学研究者定义“数字图书馆是若干联合结构的总称,它使人们能够智能地和实实在在地存取全球网络上以多媒体数字化格式存在的、为数巨大的且仍在不断增多的信息”。国外有学者认为“数字图书馆是一种有纸基图书馆外观和感觉的图书馆,而其内部全部资料已被数字化并存储起来,可在网络环境中被本地和远程用户存取,还能通过自动控制系统为用户提供先进的、自动化的电子服务”。我国有的学者认为“数字图书馆的概念是组织数字化信息及其技术进入图书馆并提供有效服务,即将图书馆所有载体的信息均能以数字化的形式被存取和管理,通过网络组织读者访问外界数字化图书馆和文献信息数据库系统”。

数字图书馆的特征是:①使用计算机技术将各种文献信息资源数字化,并提供网上服务;②通过各种通信手段和计算机网络,连接国内外的数字图书馆和各种文献信息数据库系统;③利用光盘存储、超媒体、数据仓库等各种先进技术,组织大型的数据库的管理与检索;④利用计算机等技术手段进

行电子参考咨询,为用户解决联机查找时遇到的各类问题。

数字图书馆的功能有:①各种载体信息的数字化;②数据的存储和管理;③组织对数据的有效访问和查询;④数字化资料的网上发布和传送;⑤系统的管理和版权保护;⑥系统的安全运行和可靠服务。

20世纪90年代以来,随着数据通信、计算机网络、多媒体以及高密度存储等技术的发展与应用,1993年由美国国家科学基金会(NSF)、美国国防高级研究计划署(DARPA)和国家航空与太空总署(NASA)联合提出了数字图书馆创始工程(代号DLI),1994年—1999年共投资6800万美元,其中2440万美元拨给以6所大学为中心的“数字图书馆创始工程第一期工程”,即DLI-1。1995年初美国IBM公司推出全球数字图书馆计划。1995年5月由美、英、法、日、德、加、意七国的国家图书馆组成“G7数字图书馆联盟”,其目标是以现有的数字化项目为基础,构建一个人类知识的虚拟仓库,通过网络为公众取用。“法国数字图书馆”计划的目标是实现馆藏精华数字化及网络存取,并从事数字图书馆解决方案的研究、开发和商品化。日本国会图书馆耗资4亿美元进行“关西图书馆工程”,其目标是成为日本最大的数字图书馆和亚洲地区文献提供中心,2002年完成了一期工程。近年来,国外出版了近10种专门研究数字图书馆的期刊,如:D-Lib Magazine、International Journal of Digital Library等。我国于1998年8月由文化部牵头,提出建设中国数字图书馆工程,即要建立超大规模的、可以跨库检索的中文海量数据库及其信息服务机制。1999年4月开通中华文化信息网成为数字图书馆资料库的一个运行平台。同年11月数字图书馆工程项目组进行了试点工作。此项目已被提升为国家数据库工程的示范工程,列入了国家“863”计划和“十五”计划。

(陆长旭)

shuzi tuxiang chuli

数字图像处理(digital image processing)

通过计算机对图像进行去除噪声、增强、复原、分割、提取特征等处理的过程、理论、方法和技术以及以之为研究对象的学科。

视觉是人类最重要的信息感知手段,图像又是视觉的基础,图像处理是通信、遥感、医疗、气象、军事等诸多领域的有效工具。20世纪20年代,图像

处理首次应用于改善伦敦和纽约之间海底电缆发送的图片的质量。到50年代,数字计算机应用于图像处理之后,数字图像处理开始起步。1964年美国喷气推进实验室用计算机对“徘徊者七号”太空船发回的大批月球照片进行处理,收到明显的效果。60年代末,数字图像处理具备了比较完整的体系,形成了一门新兴的学科。70年代,数字图像处理技术得到迅猛的发展,理论和方法进一步完善,应用范围更加广泛。在这一时期,图像处理主要和模式识别及图像理解系统的研究相联系,如文字识别、医学图像处理、遥感图像的处理等。20世纪70年代后期到现在,各个应用领域对数字图像处理提出越来越高的要求,促进了这门学科向更高级的方向发展。特别是在景物理解和计算机视觉(即机器视觉)方面,图像处理已由二维处理发展到三维理解或解释。近几年来,随着计算机和其他各有关领域的迅速发展,例如在图像表示、科学计算可视化、多媒体计算技术等方面的发展,数字图像处理已从一个专门的研究领域变成了科学研究和人机界面中的一种普遍应用的工具。

一般来讲,对图像进行处理(或加工、分析)的主要目的有:

(1) 提高图像的视感质量 如进行图像的亮度、彩色变换,增强、抑制某些成分、对图像进行几何变换等,以改善图像的质量。

(2) 提取图像中所包含的某些特征或特殊信息 这些被提取的特征或信息往往为计算机分析图像提供方便。提取特征或信息的过程是模式识别或计算机视觉的预处理。提取的特征可以包括很多方面,如频域特征、灰度或颜色特征、边界特征、区域特征、纹理特征、形状特征、拓扑特征以及关系结构等。

(3) 图像数据的压缩与编码 以便于图像的存储和传输。不管是何种目的的图像处理,都需要由计算机、图像获取设备、图像输出设备以及相关的软件来完成图像的输入、加工和输出处理。

数字图像处理研究的内容主要有:

(1) 图像获取和图像表示 主要是把模拟图像信号转化为计算机所能接受的数字形式,以及把数字图像按用户所需要的形式显示出来。

(2) 图像复原 当造成图像退化的原因已知时,复原技术可用来进行图像的校正(参见图像复原)。复原技术是基于模型和数据的图像恢复,其目的是消除退化的影响,从而产生等价于理想成像

系统所获得的图像。

(3) 图像增强 当无法知道与图像退化有关的定量的信息时,可以使用图像增强技术较为主观地改善图像的质量。

(4) 图像分析 对图像中的不同对象进行分割、特征提取(参见**图像分割与图像特征提取**)和表示,从而有利于计算机对图像进行分类、识别、理解或解释。

(5) 图像重建 由图像的多个一维投影重建该图像,可看成是一种特殊的图像复原技术(参见**图像重建**)。

(6) 图像压缩和编码 对图像进行编码的主要目的是为了压缩数据和组织数据,便于图像的存储、传输和应用,特别是在互联网环境下的各种应用。当前的一些编码方法对图像数据的安全和知识产权的保护也提供了越来越多的措施(参见**图像的压缩编码**)。

数字图像处理的数学工具可分为三大类。第一类包括各种正交变换和图像滤波等方法,其共同点是将图像变换到其他域(如频域)中进行处理(如滤波)后,再变换到原来的空间(域)中;第二类方法是直接在空间域中处理图像,它包括各种统计方法、微分方法及其他数学方法;第三类是数学形态学运算,它不同于常用的频域和空域的方法,是建立在积分几何和随机集合论的基础上的运算。由于被处理图像的数据量非常大且许多运算在本质上是并行的,所以**图像并行处理**也是图像处理中的主要研究方向。

数字图像处理主要应用于如下领域:

(1) 通信 包括图像传输、电视电话、电视会议。

(2) 遥感 无论是航空遥感或卫星遥感,都需要使用图像处理技术加工处理并提取有用的信息。遥感图像处理可用于矿藏勘探和森林、水利、海洋、农业等资源的调查,自然灾害预测预报,环境污染监测,气象卫星云图处理以及用于军事目的的地面目标识别。

(3) 医疗诊断 如通过 X 射线、超声、计算机断层摄影(CT)、核磁共振等进行成像,结合图像处理与分析技术,进行疾病的分析与诊断。

(4) 工业生产 主要有产品质量检测,生产过程的自动控制,计算机辅助设计与制造等。

(5) 机器人视觉 通过实时的图像处理,对三维景物进行理解和识别,主要用于军事侦察、危险环

境作业、自动流水线上装配工件的识别和定位,以及邮政、家庭服务等。

(6) 出版、广告及视频制作 包括彩色图片的绘制、编辑、加工和分色处理,电视制作中的图像特技处理和图像合成技术等。

(7) 军事、公安、档案管理等其他方面的应用 如军事目标的侦察、制导和警戒系统、自动火器的控制及反伪装,指纹、手迹、印章、人像等的处理和辨识,古迹和图片档案的修复和管理等。

参考文献

1. Castleman K R 著. 数字图像处理. 朱志刚等译. 北京:电子工业出版社, Prentice Hall 出版公司, 1998

2. 赵荣椿等. 数字图像处理导论(第二版). 西安:西北工业大学出版社, 1995 (朱志刚 张福炎)

shuzi weifen fenxiji

数字微分分析机 (digital differential analyzer)

用数字技术实现积分运算操作的电子计算机,也称**数字积分机**。其运算原理同解算微分方程的模拟计算机一样,但却没有高放大倍数的直流运算放大器难以避免的零点漂移等问题,从而具有更高的稳定性。数字微分分析机借鉴机械式微分分析机(20 世纪 30 年代初由 Vannevar Bush 等在美国麻省理工学院研制成功,用于求解常微分方程)的设计思想。其原理如图 1 所示。设大圆盘以角速度 $d\alpha/dt$ 转动。与大圆盘相垂直的小圆轮因摩擦力而随之转动,设其角速度为 $d\beta/dt$ 。又设小圆轮与大圆盘的接触点距大圆盘轴心的距离为 y ,且可调节改变;小圆轮半径为 r 。可得以下传动关系:

$$d\beta/dt = (y/r)(d\alpha/dt) \quad (1)$$

由此可导出:

$$\beta = \beta_0 + \int_{\alpha_0}^{\alpha} \frac{y}{r} d\alpha \quad (2)$$

数字微分分析机的积分原理实质上与上述原理

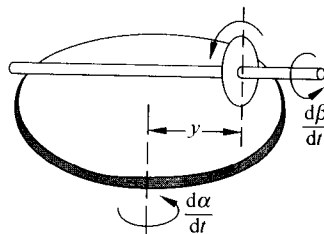


图 1 机械积分原理图

一样,只是用数字电路取代大圆盘、小圆轮等传动装置;用电脉冲代表单位增量;用脉冲的重复频率代表角速度,如图2所示。图中R、Y是两个位数相同的计数器(通常是二进制)或移位寄存器。Y中可设

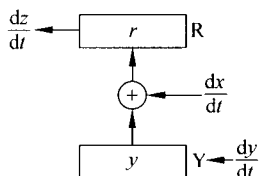


图2 数字积分原理图

置1个等于或小于 2^n 的数 y , $y = 2^n/m$,其中比例数 $m \geq 1$ 。 y 在重复频率(速度)为 dx/dt 的脉冲控制下,累加到R中。当加到使R中的数 $r = 2^n$ 时,便产生1个进位(溢出)脉冲。于是,R的进位脉冲的重复频率 dz/dt 与 dx/dt 之间存在着如下分频关系:

$$\frac{dz}{dt} = \frac{1}{m} \frac{dx}{dt} = \frac{y}{2^n} \frac{dx}{dt} \quad (3)$$

由此可导出:

$$z = z_0 + 2^{-n} \int_{x_0}^x y dx \quad (4)$$

对照式(1)可见,式(3)中的 dz/dt 相当于小圆轮角速度 $d\beta/dt$, dx/dt 则相当于大圆盘的角速度 $d\alpha/dt$, y 值正好对应于小圆轮离大圆盘轴心的可变距离。 y 可以随加(或减)计数脉冲的重复频率(dy/dt)而变。把这样的数字积分器和其他数字电路按问题的数学描述(即常微分方程)连接起来(参见模拟计算机),就组成了能连续、实时求解这一常微分方程的数字微分分析机。

数字微分分析机在20世纪50年代曾获广泛应用,尤其用在军事部门。后来,因通用数字计算机也具有求解微分方程的强大而灵活的功能,专用的数字微分分析机被逐渐淘汰。

参考文献

Шилейко А. В. Цифровые Дифференциальные Анализаторы. Москва: ВИНТИ, 1961 (童颖)

shuzi xitong luoji zonghe jishu

数字系统逻辑综合技术 (logic synthesis technologies for digital system) 数字系统的逻辑设计及其优化技术。数字系统的逻辑设计的给定条件包括逻辑功能及逻辑实现的约束。逻辑功能由硬件描述语言(HDL)或布尔方程等各种形式给

出。而逻辑实现的约束指逻辑单元的类型及其工艺条件等。逻辑综合的目标则是获得一个优化合理(成本最低或接近最低)的方案,即将一个用HDL给出的并经核实有效的设计描述,转换成由逻辑门构成的优化合理的网表描述,并同时产生一个用于验证和检测逻辑功能的完全测试集。

逻辑综合技术是计算机设计自动化技术的一个重要组成部分。在逻辑综合的早期研究领域中,采用手工设计的方法来减少电路中门的数目,分别有布尔代数法、卡诺图法和真值表化简法(也称Quine-McCluskey法)。这些方法的致命弱点是无法综合规模较大的逻辑电路,输入变量不能多于6~7个。逻辑综合包括组合逻辑电路的综合和时序逻辑电路的综合。由于数字系统规模越来越大,功能更加复杂,而要求设计周期短,这就驱使人们采用计算机辅助设计的方法,即自动逻辑综合技术,完成整个系统的逻辑综合。超大规模集成电路的逻辑综合的目标有:芯片总布图面积最小化、关键路径时延最小化、可测试性最大化并辅助提供一个完备测试集。

数字系统的逻辑综合分为系统级和逻辑级。

系统级逻辑综合主要包括硬件描述语言编译、数据流综合和控制流综合等部分。编译是指自动将数字系统的功能描述转换成数据流图和控制流图;数据流综合主要根据数据流图,选择寄存器、多路选择器和存储单元等逻辑元件,将其综合成数据流的逻辑图;控制流综合则根据控制流图以及数据流综合对控制部分的要求,综合产生控制流的逻辑图,同时给出HDL逻辑描述的设计文本提供给逻辑级的综合。

逻辑级的逻辑综合则主要是将一个逻辑电路的HDL验证正确的设计描述,转换成由逻辑门构成的网表描述,该级逻辑综合应包括:①逻辑描述及其编译 将数字系统的逻辑功能描述转换成逻辑级的中间描述形式或逻辑电路的描述形式。②逻辑划分

将所要设计的逻辑电路如何有效合理地分解成多个逻辑子电路,便于整体上更好处理并获得更优的结果。③状态化简 是在不改变系统性能前提下,将逻辑电路状态表的内部状态数减少到最少或接近最少的过程。④状态分配 又称状态编码。就是将状态表中的各状态名分别赋予一个二进制值,使得状态表(即功能描述)可以用布尔表达式或方程组来表示。状态分配的好坏,直接决定了逻辑电路组合部分的复杂度。⑤逻辑优化 分两级和多级逻辑

优化。两级逻辑优化主要是指可编程逻辑阵列的逻辑最小化。它是在不改变逻辑功能的条件下,采用因子提取、无关项合并等方法,寻求一个具有最少变量和最少乘积项数的逻辑表达,从而降低组合逻辑的复杂度。多级逻辑优化则侧重于采用适当增加逻辑电路级数的办法来降低复杂度,简化原电路,而且不改变其逻辑功能和限制条件,降低造价。它通常与状态化简和状态分配结合在一起考虑。

参考文献

刘明业主编. 数字系统设计自动化. 北京: 电子工业出版社, 1991 (董云耀)

shuzi xitong moni jishu

数字系统模拟技术 (simulation technologies for digital system)

对给定的数字系统在计算机上建立模型来模拟其行为、功能或性能的技术。它具有预测和验证双重意义。模型的建立和输入主要包括模拟对象各组件的功能特性、互连关系或相互作用关系、外部激励信号等描述信息,而这些信息通常是由专门的硬件描述语言(如 VHDL)来描述提供。

根据模拟对象和模拟层次的不同,模拟主要分为两类,即电路模拟和逻辑模拟,其他还有系统模拟、器件模拟等。以下介绍电路模拟和逻辑模拟。

电路模拟 又称电路分析。主要用于小规模数字系统的性能分析。自 20 世纪 60 年代起,人们就致力于计算机辅助电路模拟技术的研究。1962 年出现了第一个通用电路分析程序。集成电路进展所提出的需求推动了电路分析技术的发展,陆续出现了许多实用的软件。其中最著名的是美国加州大学伯克莱分校开发的 SPICE 程序。这类程序的基础是关于电路的电压、电流和电阻的矩阵方程求解问题。其特点是精度很高但运算时间很长。优秀的模拟算法能够给出精确的直流特性、温度、定时、频率、灵敏度和噪声分析。其中定时分析主要是指电路的时延、竞争和冒险分析,在数字系统规模很大时尤为重要。于是,就出现了专门用于定时分析的软件。典型的如美国的 MOTIS 程序,它是在模型处理(例如允许门和晶体管作为模拟的基本单元同时存在)和数值方法上作了改进。它的求解速度比 SPICE 几乎快两个数量级,但这是以降低精度(约 10%)为代价的。电路模拟程序和定时分析程序两者在精度和时间上各有优势,在实际使用时往往把两者结合起来。

对那些影响整个网络特性的关键部分或临界通路部分就采用电路模拟程序进行局部的、详尽的分析,而对整个电路则采用定时分析程序来分析。

逻辑模拟 主要目的有两个:一是检查逻辑设计的正确性;二是进行故障模拟,产生故障诊断的测试码。按照组成的基本逻辑单元的规模大小,逻辑模拟可分为寄存器传输级、功能块级、门级和开关级逻辑模拟。寄存器传输级的基本单元是寄存器、存储器、总线和算术逻辑运算单元等系统组件;功能块级的基本单元是锁存器、触发器和计数器等功能部件;门级的基本单元是非门、与门、或门等各种逻辑门;开关级则将逻辑门进一步展开成晶体管,每个晶体管相当于一个开关,同时考虑晶体管导通电阻和节点电容的影响。开关级模拟常常作为介于逻辑模拟和电路模拟之间的一种单独的模拟方式。

随着电路规模和复杂度的提高,人们希望有一种混合的模拟程序能把逻辑模拟和电路模拟结合起来,以适应对模拟的精度和速度的不同要求。美国的 SPLICE 程序是这类混合模拟程序的典型代表,它允许晶体管、门甚至功能块同时作为模拟的基本单元。选用何种模拟形式取决于设计对模拟对象、精度和时间的综合考虑。

模拟技术的最新进展是针对高速集成电路进行的定时验证技术。这是一种动态时序分析技术,是传统逻辑模拟技术的发展。它通过分析线路设计的全部时序组合,提供最坏情况的时序信息,即描述出信号的不定值区域。模拟技术的进一步发展将是各种不同的模拟手段相互补充、不断完善的过程。

参考文献

刘明业主编. 数字系统设计自动化. 北京: 电子工业出版社, 1991 (董云耀)

shuzi xiangji

数字相机 (digital camera) 采用数字方式存储所拍摄图像的照相机。又称数码相机。

数字相机用一组与普通照相机相同的光学镜头来成像,然后用一种被称为固体图像传感器的光敏半导体器件来获取图像。这种固体图像传感器是一种光敏芯片,它不仅对光作出反应,把光信号转换成电信号,而且通过其中的模拟数字转换器把信号转换成数字信号,按一定的格式将其压缩保存在相机内部或外接的存储设备中。通过接口,所存储的图像可以传输到计算机中保存、显示、处理和打印。数字相机还可以直接连接到专用的打印机上输出照片。

数字相机一般都带有一个小巧的液晶屏来观察拍摄场景和观看拍摄的相片,部分数字相机还支持连续视频图像拍摄。

衡量数字相机的主要技术指标如下:

(1) 分辨率 单位为像素。常用的分辨率有 256×256 像素, 640×480 像素, $1\,600 \times 1\,200$ 像素, 以及 $2\,400 \times 2\,000$ 像素。分辨率越高, 图像质量越好。

(2) 固体图像传感器类型 目前主要有电荷耦合器件 (CCD) 和互补金属-氧化物-半导体 (CMOS) 两种图像传感器。CCD 的性能好, 但价格和功耗较高, 用于高档数字相机。CMOS 的性能比 CCD 差, 但价格便宜, 用于普及型相机。图像传感器以英寸为单位, 常见的有 $1/2$ 英寸、 $1/3$ 英寸等, 尺寸越大拍摄效果越好。由于图像传感器只对光强响应, 所以在拍摄彩色照片时, 先用分光系统将光线分成三原色, 再由图像传感器接收转换, 然后用数字方式合成还原。

(3) 存储媒体类型和容量 数字相机一般用闪存芯片作存储器, 有 SmartMedia, CompactFlash 和 Memory Stick 等几种形式, 存储容量从 8 MB 到 1 GB 不等。部分数字相机支持可擦写光盘或者微型硬盘, 容量可达到 2GB 以上。

(4) 图像存储格式 数字相机支持 JPEG 或 Tiff 格式。JPEG 格式的压缩比大, 节省空间, 图像质量不如 Tiff 格式好。Tiff 格式是无损压缩, 但占用较多的空间。

(5) 输出接口 主要有 USB 接口和 RS-232 串口。USB 接口是数字相机的标准配置, 用于和计算机相连; 串口速度较慢, 并需要专门的驱动软件。

(6) 微调焦系统 为了拍摄极近距离的物体, 有些数字相机配有对镜头进行微调焦的系统。

数字相机产生于 20 世纪 70 年代。美国由于军事上的需要, 用卫星在空中对目标进行拍照, 推动了数字相机技术的产生与发展。20 世纪 90 年代伴随着数字影像与计算机技术的飞速发展, 数字相机开始进入民用市场, 1994 年诞生了第一款消费型数字相机。随着固体图像传感器技术和半导体存储技术的发展, 数字相机迅速普及, 图像质量也越来越高。通过高质量的打印机, 图像输出质量已可与普通照片相比。数字相机不仅可以取代传统采用胶片的照相机, 并已成为一种新型的计算机图像输入设备。

参考文献

1. (美) Bill Erickson, Frank Romano 著. 数字相机与数字摄影技术. 承健, 张耀炽, 玄立勋等译. 北

京: 电子工业出版社, 1999

2. 祖厚. 数字照相机. 上海: 上海科学技术文献出版社, 2002 (谢长生 黄浩)

shuzi xinhao chuliqu

数字信号处理器 (digital signal processor, DSP) 适合数字信号处理的专用处理器。随着电子技术的发展, 数字信号处理器常以大规模集成电路或超大规模集成电路芯片的形式来实现, 称为 DSP 芯片。DSP 芯片可以高速地实现过去由软件实现的大部分数字信号处理算法 (如相关、卷积、滤波、快速傅里叶变换 (FFT) 等)。

数字信号处理器大体上可分为两类: 一类是专门为实现某种数字信号处理算法而设计的, 仅适用于某些专门领域, 称为专用数字信号处理器; 另一类是可编程的, 可以实现各种数字信号处理算法, 从而可满足多个领域的需求, 称为通用数字信号处理器。实现某种数字信号处理算法的 DSP 芯片将算法固化在其内部结构之中。例如专门实现 FFT 算法的 FFT 处理器, 专门处理相关运算的数字相关器。又如英国 INMOS 公司生产的 A100 是专门用于处理一维滤波的 DSP 芯片, 适合处理雷达、声纳和卫星通信系统的高速数据流; 该公司的另一产品 A110 可高速完成一维和二维卷积运算, 适合二维图像处理。这类数字信号处理器本身不需要编程, 就能高效地实现数字信号处理算法。在实际应用时, 将多个 DSP 芯片和微处理器 (或计算机) 以及其他器件或设备连接成一个系统, 其中, DSP 芯片可高效实现数字信号处理算法, 而微处理器则起控制和管理作用。

可编程的数字信号处理器的体系结构有一系列特点。由于数字信号处理的主要算法都包含大量的乘法和加法, DSP 芯片内的乘法器和加法器可在 1 个时钟周期内完成 1 次乘法和 1 次累加, 这种乘加运算还可不间断地连续执行。数字信号处理器采用哈佛结构, 即程序和数据分别存放在独立的存储器中, 并有独立的程序总线 and 数据总线, 这允许取指令和执行指令在时间上完全重叠。处理器的指令系统除了包含程序存储器和数据存储器之间的数据传送指令外, 还能满足各种数字信号处理算法的特殊要求, 例如相关运算中的连续乘加、FFT 运算中的倒序等。为了满足连续乘加等指令对数据量的需求, DSP 芯片中有多个地址生成器, 可在 1 个时钟周期内生成多个存储器地址。另外, DSP 芯片还可将程序的循环机制用硬件实现, 以减少开销。总之, 数字

信号处理器使多种操作并行执行,使数字信号处理算法得以高效实现。

可编程的数字信号处理器在 20 世纪 80 年代初期已经得到了广泛使用。后来在芯片中增加了浮点乘加器,提高了运算精度和运算速度。90 年代以来,DSP 芯片都能提供多个高速通信口,可以方便地将多个这样的芯片组成多处理机并行处理系统。例如美国 TI 公司于 1994 年推出的 TMS 320C40 有 6 个 DMA 通信口,可构成三维的阵列处理系统。数字信号处理器已广泛用于图形与图像处理、语音信号处理、声纳与雷达信号处理、地震资料处理、数字通信、智能机器人等。随着日益增长的应用需求和微电子技术的发展,DSP 芯片上包含了更多的并行部件。例如,TI 公司于 1996 年推出的 TMS 320C80 有 4 个定点运算器和 1 个浮点运算器,它可以在 1 个周期内完成 5 条指令的执行和 10 个不相关操作数

的读取。其峰值速度达每秒 20 亿次操作。2004 年推出的 TMS 320C64X 可在 1 个周期内完成 8 个乘加操作,其峰值速度达 80 亿次每秒。

(侯朝煊 钟玉琢)

shuzi yinpin bianji

数字音频编辑 (editing digital audio) 采用专门的软件对数字音频信息进行各种编辑处理以达到使用要求的过程和技术。

在制作多媒体文档时,人们越来越多地需要自己录制和编辑数字声音(如背景音乐、解说词等),目前多半是在个人计算机(PC机)上使用音频编辑软件完成的。音频编辑软件有多种,它们能方便直观地对数字声音(wav 文件)进行各种编辑处理(图 1),通常包括如下一些功能:

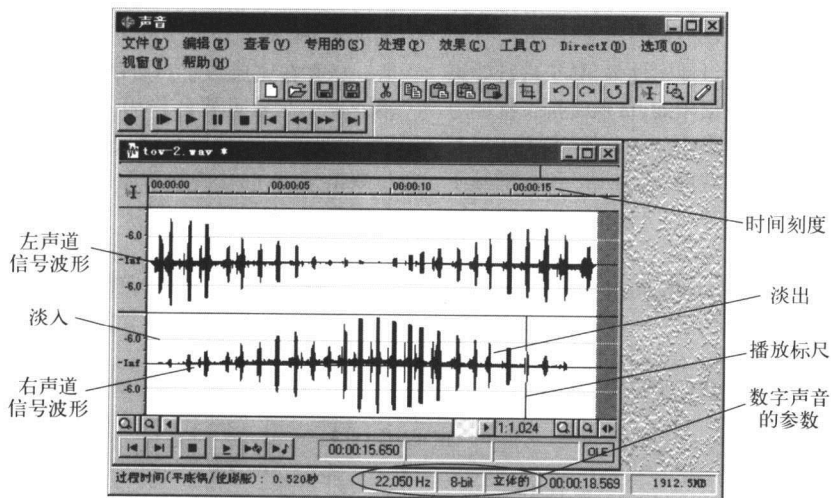


图 1 声音编辑软件的界面

(1) 基本编辑操作 例如声音的剪辑(删除、移动或复制一段声音,插入空白等),声音音量调节(提高或降低音量,淡入、淡出处理等),声音的翻转,持续时间的压缩、拉伸,消除噪声,声音的频谱分析等。

(2) 声音效果处理 包括混响、回声、延迟、频率均衡、和声效果、动态效果、升降调、颤音等。

(3) 格式转换 如将不同取样频率和量化精度的波形声音进行转换,将不同文件格式的波形声音相互转换,将存储波形音频文件(wav)格式声音与运动图像专家组第 3 层音频文件格式(MP3)声音相

互转换,将 wav 音乐转换为乐器数字化音乐等。

(4) 其他功能 如分轨录音,为影视配音,刻录 CD 唱片等。

参考文献

张福炎,孙志挥. 大学计算机信息技术教程. 南京:南京大学出版社,2003 (张福炎)

shuzi yinpin huoqu

数字音频获取 (acquisition of digital audio)

使用特定的设备将声音信号经过数字化处理后输入计算机、数字录音笔或数字摄像机等数字设备的

过程。

声音是振动产生的波,是一种模拟信号。为了使用计算机等数字设备进行存储、处理和传输,首先必须将它转换成数字表示形式。声音信号数字化的过程如图1所示。取样的目的是为了把时间上连续的信号转换为时间上离散的一组样本。根据取样定理,取样频率不应低于声音频谱中最高信号频率的两倍。因此,语音信号的取样频率一般为8 kHz,高

保真音乐的取样频率应在 40 kHz 以上。量化实际上就是进行模数转换,即把每个样本用二进制整数来表示。声音信号的量化精度一般为 8 位、12 位或 16 位,量化精度越高,声音的保真度就越好。经过取样和量化后的声音,还必须按照一定的要求进行编码,即进行数据压缩,以减少数据量,并按某种格式将数据进行组织,以便于计算机等数字设备存储和处理,或者在网络上进行传输。

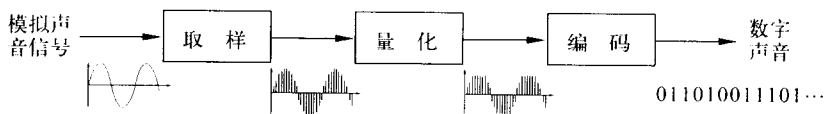


图 1 声音信号的数字化

多年来,声音信号的记录、回放、传输、编辑等一直是使用模拟装置进行的。把模拟声音信号转变成数字形式并使用数字设备进行声音的处理有许多优点。例如,以数字形式储存的声音回放性能好,复制时没有失真;数字声音的可编辑性强,易于进行效果处理;数字声音能进行数据压缩,传输时抗干扰能力强;数字声音容易与其他媒体(如文本、图像)相互结合;它也为自动提取声音的“元数据”和实现基于内容的检索创造了条件。

计算机获取数字声音一般是使用声卡(有些计算机声卡与主板集成在一起)完成的。声音可以从话筒(麦克风)输入,也可以从线路(音响设备或 CD 唱机等)输入。声卡以数字信号处理器(DSP)为核心。DSP 在完成数字声音的编码、解码并在声音编辑处理中起着重要的作用。图 2 是声卡的原理框图,它通过小计算机系统接口(PCI)总线与主机进行数据交换,混音器的功能是将不同的声音信号进行混音,并实现音量控制。

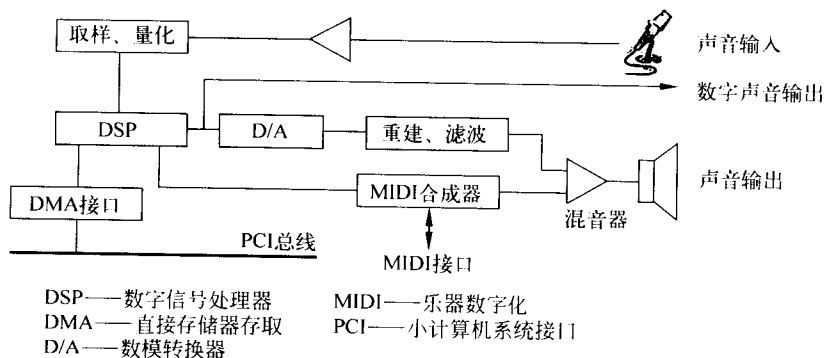


图2 声卡的原理图

除了利用声卡进行在线声音获取之外,也可以使用数码录音笔进行离线声音获取,然后再通过通用串行总线(USB)接口直接将已经数字化的声音数据送入计算机中。数码录音笔的原理与上述过程基本相同,不过由于带宽的原因,它一般仅适合于录制语音。

参考文献

张福炎,孙志挥. 大学计算机信息技术教程.

南京:南京大学出版社, 2003

(张福炎)

shuzi yonghu zhuanrongxian

数字用户专用线 (digital subscriber line, DSL) 在现有的基于铜缆的提供电话服务的线路上传输高速数据的用户接入技术。主要有以下三类: **高比特率数字用户专用线 (HDSL)**、**不对称数字**

用户专用线(ADSL)、甚高速数字用户专用线(VHDSL),统称为XDSL。在光纤接入网尚未普及时,用户迫切需要一种能够利用现有双绞线提供宽带业务接入的技术。XDSL接入技术可以在现有铜线用户专用线上经济有效地进行短距离数字通信,方便地实现宽带业务。

(1) 高比特率数字用户专用线(HDSL) 是在无中继的用户环路上使用无负载电话线提供高速数字接入的技术,典型速率2 Mb/s,可以在任意双绞线实现高带宽双向传输,距离达3~5 km,不需选择线对,采用线路码,误码率低。HDSL技术已经发展得比较成熟,可以为用户提供30B+D,2 Mb/s租用线,也可用来提供30路语音(言语)。可对家庭和商业用户提供点播电视、交互信息、电视购物和远程教育等交互视频服务。

(2) 不对称数字用户专用线(ADSL) 是在无中继的用户环路上使用有负载电话线提供高速数字接入的技术。ADSL的速率是不对称的,其下行数字信道速率可达6 Mb/s,上行数字信道速率为144 kb/s或384 kb/s,可在现有任意双绞线上传输。它采用线路码,误码率低,用户的模拟话路独立。ADSL对少量使用宽带业务的用户是一种经济快速的接入方法,它有为不同速率的业务提供服务的能力,如分别提供MPEG-1(运动图像专家组标准)质量的图像传送、交互式信息传输和视频服务以及宽带信息服务。

(3) 甚高速数字用户专用线(VHDSL) 是在ADSL基础上发展起来的,但采用了不同的编码方案,在很短的双绞线上可传送比ADSL更高速的数据,当传输线长度不超过300 m时,其最大的下行速率可达51~55 Mb/s,当传输速率在13 Mb/s以下时,传输距离可达1.5 km。上行速率也提高到1.6 Mb/s以上。VHDSL上下行速率也是不对称的,下行速率有三档:13 Mb/s、26 Mb/s、52 Mb/s,相应的传输距离为1.5 km、1 km和0.3 km。上行速率也有三档:1.6 Mb/s、2.3 Mb/s、19.2 Mb/s。VHDSL仅处于研究阶段,尚无国际标准。

随着Internet应用的扩大,用户对宽带的要求越来越高。虽然发展光纤用户网是今后主要方向,但不可能在短期内实现。因此采用XDSL技术,满足高速短距离传输业务需求仍是今后一段时期的发展策略。XDSL的典型应用有:①Internet高速接入;②视频点播、交互电视和网上购物等宽带多媒体服务;③远程LAN接入、异地办公及在家办公等

高速数据应用;④远程医疗、远程教学、视话会议;⑤专用网的应用。ADSL特别适用于用户密度低的居民区和地理上分散的小企业或部门。HDSL作为一种能有效利用现有用户网资源进行短距离高速数据传输的手段,在用户网光纤化的过渡期间将会发挥重要作用。

参考文献

1. 胡道元编著. 智能建筑计算机网络工程. 北京:清华大学出版社,2002
2. Padmanand warrier, Balaji Kumar, XDSL Architecture. McGraw-Hill Inc., 2000 (胡道元)

shuzu leixing

数组类型(array type) 分量类型均为同一类型的一种构造类型。实际问题中常遇到向量、矩阵之类数据结构,在高级语言中抽象成数组类型,一般写成如下形式: $M: \text{array } D \text{ of } R$

这里 D 为数组 M 的下标类型, R 为数组 M 的分量类型。

从抽象的观点来看,一个数组可看成为由下标类型 D (定义域)到分量类型 R (值域)的映射。如果令 $i \in D$,则 $M[i]$ 表示对应 i 的数组分量。

数组类型的分量类型可以是某种构造类型,特别当其分量类型是数组类型时,这种数组称为多维数组。数据元素的数目由其下标类型的取值范围决定,如果数组的所有下标类型的取值范围可在编译时确定,那么这种数组称为静态数组,否则称为动态数组。从编译的角度考虑,静态数组可在编译时分配它的存储空间,而动态数组要到执行时刻才能分配它的存储空间。

字符串类型在有的高级语言中被定义为以字符类型为分量类型的一维数组,其下标类型用最小值为1,最大值大于1的子域类型定义。(陈涵生)

shuangjiaoxian dianlan

双绞线电缆(twisted pair cable) 一种外部包裹屏蔽或塑橡胶皮,按一定密度的螺旋结构排列的两根绝缘铜线组成的传输介质。双绞线在早期多用于电话等传统的通信工程。在计算机通信的早期阶段,点对点通信也采用双绞线通信。随着技术的进步,双绞线通信可支持的传输速率不断提高,因而在计算机网络系统中的使用越来越广泛。

双绞线根据是否具有屏蔽性能分为屏蔽双绞线

(STP) 和非屏蔽双绞线 (UTP) 两类, 如图 1、图 2 所示。

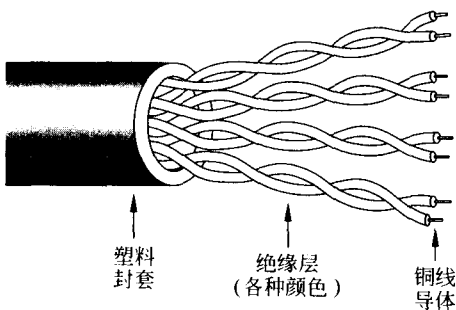


图 1 非屏蔽双绞线(UTP)

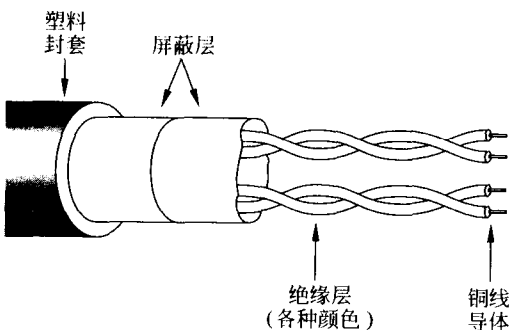


图 2 屏蔽双绞线(STP)

屏蔽双绞线内有一层金属薄膜作为保护, 可以减少当信号传送时所产生的电磁干扰, 相对的价格比非屏蔽双绞线贵。

非屏蔽双绞线内无金属膜, 所以对电磁干扰的敏感性相对较大。

双绞线根据传输速率的高低分为三类、四类、五类、六类双绞线。

(王晓东)

shuiping zixitong

水平子系统 (horizontal subsystem) 结构化布线系统中连接用户工作区与布线系统主干线的子系统。水平子系统由每层配线间至信息插座的配线电缆和工作区用的信息插座等组成, 如图 1 所示。

在结构化布线系统中, 水平子系统起着支线的作用, 它将所有用户端子通过一些连接件连接到配线设备上。

在设计水平子系统时, 除了根据不同的应用需求选择不同类型、不同颜色的双绞线电缆外, 还应根据安装地点的环境, 选择好布线的路由及布线方式。

水平布线系统中常用的电缆类型有:

- (1) 100Ω 非屏蔽双绞线(UTP)电缆;
- (2) 100Ω 屏蔽双绞线(STP)电缆;
- (3) 50Ω 同轴电缆;
- (4) 62.5/125μm 光纤电缆。

其中, 双绞线电缆和光缆最为常用。

水平布线路由及方式受到很多因素的影响, 常用的布线方式, 大致有两种, 一种是地板下直接埋管布线法, 如图 2 所示。它采用星状结构, 利用金属线槽或金属管从布线系统的配线间直接引到每个信息点。

另一种是线槽管道布线法。通常是在天花板内安装线槽, 再用管线从线槽引到每个插座, 如图 3 所示。

在选择布线路由时, 只要有可能, 就应使电缆隐藏在天花板或地板内, 如果暴露在外, 要保证电缆排列整齐, 并尽可能将电缆布在墙角或其他隐蔽处。

水平布线路由还应尽量避免电磁干扰, 和强电电缆保持一定的距离, 表 1 是推荐的通信电缆与其他干扰源的间距要求:

在新建建筑中, 布线系统的设计应在设计大楼的图纸时考虑, 并融进大楼的弱电图中, 以便在施工过程中暗敷相关的线槽和管线, 预留墙面出口, 安装插座底盒。

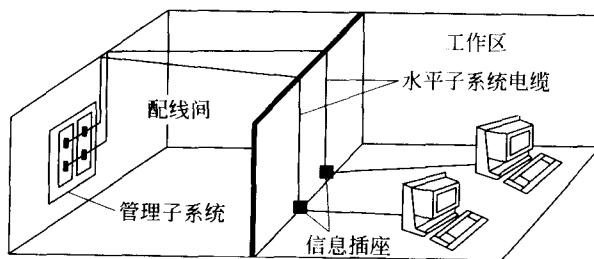


图 1 水平子系统

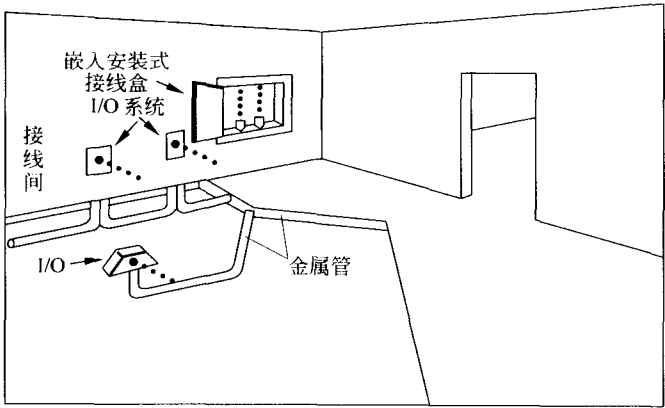


图2 直接埋管布线法

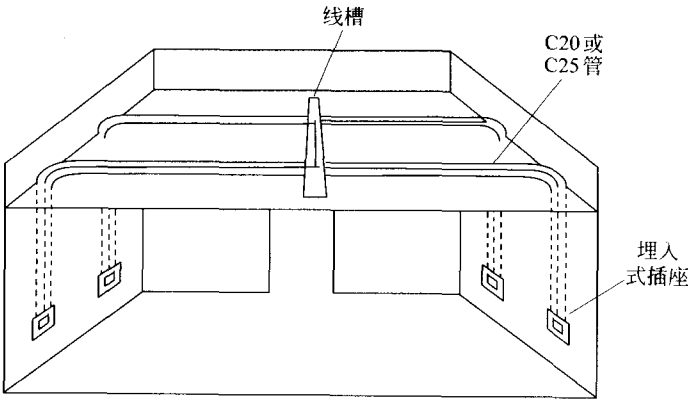


图3 线槽管道布线法

表1 通信线与电力线间距表

其他干扰源	与综合布线接近状况	最小间距 (cm)
380V 以下电力电缆 < 2kVA	与电缆平行敷设	13
	有一方在接地的线槽中	7
	双方都在接地的线槽中 (平均长度小于 10m)	1
380V 以下电力电缆 2 ~ 5 kVA	与电缆平行敷设	30
	有一方在接地的线槽中	15
	双方都在接地的线槽中	8
380V 以下电力电缆 > 5 kVA	与电缆平行敷设	60
	有一方在接地的线槽中	30
	双方都在接地的线槽中	15
荧光灯、氙灯、电子启动器或交感性设备	与电缆接近	15 ~ 30
无线电发射设备 雷达设备 开关电源、电磁感应炉等其他工业设备	与电缆接近	≥150
配电箱	与配线设备接近	≥100
电梯、变电室	尽量远离	≥200

参考文献

胡道元主编. 网络设计师教程. 北京: 清华大学出版社, 2001
(王晓东)

shunxu chengxu sheji

顺序程序设计 (sequential programming)

编写顺序程序的方法与过程。顺序程序的特点是其组成部分依次执行。其基本成分有顺序控制和数据控制两个方面。

顺序控制结构一般由语言文本定义了执行顺序。多数语言都将程序中语句的自然排列顺序定义为语句的执行顺序。程序人员可利用某些语句(如 GOTO 语句)改变语句的自然排列执行顺序。

顺序控制大致可分为三类: 表达式内部(以及语句内部)的顺序控制, 语句间的顺序控制, 子程序(函数、过程)间的顺序控制。数据控制大多涉及命令式语言的基本特征——变量及其绑定。

变量是一个四元组, 由变量名、变量属性、变量的值和变量的地址组成。由 W. Barron 给出的变量定义图, 如图 1 所示。

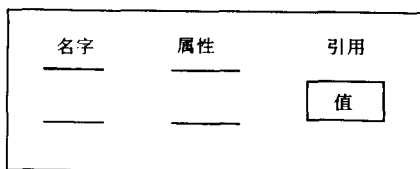


图 1 变量的定义图

变量名用来标识和引用变量。变量的重要属性有其类型、作用域和生存期。变量的值和属性在运行过程中是可能改变的。将变量与值、属性联系起来, 称为绑定。赋值语句中运算项的绑定时刻和表达式中的函数副作用, 可能影响值的惟一性。

与子程序有关的数据控制问题主要表现在两方面:

(1) 进入子程序的数据控制问题在子程序体内可能出现三类名字: 局部变量名、全局变量名和形式参数名。

过程调用的结果是通过变更其非局部环境完成计算, 而函数调用的结果只是回送一个值, 并不改变其非局部环境。函数副作用是由于函数执行过程中修改了其非局部环境或变量参数而引起的。

子程序调用时的实在参数求值及参数传递时的

数据控制问题比较复杂。实在参数在求值之前必须先检查它与形式参数在个数、顺序、类型诸方面是否一致。参数传递有四种典型的方式: ①按值调用; ②引址调用; ③按值-结果调用; ④按名调用。与数据控制有关的还有参数的引用方式, 共分三类: 只读型、只写型和读写型。

(2) 从子程序回送的数据控制问题(结果回送)

共有三种方式: ①若为函数, 可将结果值存入函数名这一特殊变量后回送, 或通过返回语句回送结果; ②利用传址参数, 或传值-结果参数回送结果; ③通过非局部变量的副作用回送结果。

参考文献

1. 郭浩志主编. 程序设计语言概念. 长沙: 国防科学技术大学出版社, 1989

2. PRATT T W. Programming Language: Design and Implementation. Englewood Cliffs: Prentice - Hall Inc., 1975
(郭浩志)

shunxu kongzhi

顺序控制 (sequential control) 使生产机械或生产过程按规定时序顺序动作, 或在来自受控对象输入信号作用下按预定规律顺序动作的控制方式。构成的系统可以是开环的, 也可以是闭环的(参见计算机控制)。

顺序控制的发展历史可以追溯到中世纪以前的各种顺序动作的机械, 以后大致经历了从机械凸轮(或转鼓)式控制、继电器-接触器控制、电子逻辑元件控制(包括数字集成式控制和矩阵式控制等)直到计算机顺序控制这样几个阶段。在以计算机为系统核心之前的各阶段, 都用硬件来改变控制逻辑, 只能服务于专用对象。而计算机顺序控制的通用性强, 可以通过软件变化完成不同的控制任务。20 世纪 70 年代出现的可编程控制器是专用于各种工业过程顺序控制而设计的微型计算机设备。

典型的顺序控制系统由顺序控制器、输入接口、输出接口、显示报警部分等组成, 它的执行机构通常是开关式(或称位式)的执行器。控制器的工作原理可以用有限自动机模型加以描述, 即控制器的当前状态和输入信号决定了输出信号, 同时也决定了控制器的下一个状态。输入信号可以由时序发生器产生的一连串信号, 也可以是各种随机

事件。

顺序控制广泛地应用于生产自动线控制、机床控制、运输机械控制、水处理设备控制等工业自动化系统中。在连续生产过程装置中,也有一些操作环节如设备启动、运输、检验、包装等可用可编程控制器实现顺序控制,并通过数字通信网络与整个过程装置的计算机控制系统连成一体。

参考文献

1. 黄一夫主编. 微型计算机控制技术. 北京: 机械工业出版社, 1988
2. 李九龄, 翁樟等. 数字控制系统. 北京: 机械工业出版社, 1988 (赵鹏程)

shuohuaren shibie

说话人识别 (speaker recognition) 由计算机利用语音信号中所包含的反映说话人生理和行为特征的语音参数自动识别说话人身份的过程和技术。又称声纹识别。

对说话人(声纹)识别的研究可以追溯到 20 世纪 40 年代, 1941 年 Bell 实验室研制成功语谱图仪, L. G. Kersta 通过观察语谱图进行识别, 提出了“声纹”的概念。60 年代, 同是 Bell 实验室的 S. Pruzansky 提出了基于模式匹配和概率统计方差分析的说话人识别方法, 出现了说话人自动识别研究的高潮。其间的工作主要集中在各种识别参数的提取、选择和实验上, 并将倒谱和线性预测(LP)等方法应用于说话人识别。20 世纪 70 年代末至今, 说话人识别的研究重点转向对各种声学参数的线性或非线性处理以及新的模式匹配方法上, 如动态时间规整(DTW)、主成分分析(PCA)、隐马尔可夫模型(HMM)、人工神经网络(ANN)以及多特征组合等技术。

如今, 说话人识别技术已逐渐进入实际应用, 但仍存在许多问题需要解决, 其中关键的特征表示问题(语音信号中哪些特征或其变换用来描述说话人是有效而可靠的)仍未能得到很好解决。

语音由发声器官运动产生。发声器官包括喉、声道、嘴和鼻。气流经过喉中的声门, 引起声带的周期性振动, 形成周期性的脉冲串进入声道, 经过声道的谐振以后, 通过嘴或鼻孔向外辐射。不同说话人的发声器官具有不同的特性, 例如声道的形状不同、嘴的开合度不一样等, 另外, 发声速度、韵律以及口音等也因说话人不同而存在差异。这些特定的说话人信息包含在语音中构成了说话人识别的物理

基础。

说话人识别可以看作是言语(语音)识别的一种。但是, 与言语识别不同的是, 说话人识别希望从语音信号中提取出说话人的特征, 而不考虑语音信号的语义内容。也即说话人识别强调说话人的个性, 试图挖掘出包含在语音信号中的个性因素; 而语音识别则企求从不同人的语音信号中寻找共同因素, 强调共性。

说话人识别的基本原理框图和生物特征识别几乎一致(参见生物特征识别), 主要包括两大功能模块: 特征提取和模式匹配。

特征提取是指从经过预处理的语音信号中提取出能惟一表现说话人身份的有效而稳定可靠的特征。目前说话人识别系统主要依靠语音的低层次声学特征进行识别, 这些特征主要包括线性预测系数及其派生参数、由语音频谱导出的参数、反映听觉特性的参数以及上述参数的组合等。

模式匹配是将识别时的特征模板或模型与训练时得到的模板或模型作相似性匹配, 计算它们之间的相似性(距离)。模式匹配的主要技术有动态时间规整(DTW)、向量量化(VQ)、隐马尔可夫模型(HMM)、人工神经网络(ANN)、高斯混合模型(GMM)、支持向量机(SVM)等, 也可以将上述多种方法与不同特征进行有机组合来提高说话人识别的性能。

说话人识别可以分为两个范畴, 说话人辨认和说话人确认。①说话人辨认 根据一段语音确定说话人是否已经注册, 并判断其身份。②说话人确认 根据说话人的语音确定该说话人是否与他所声称的身份一致。

根据用户在使用系统时提供语音的发音材料, 说话人识别可以分为与文本有关的、与文本无关的、文本提示的 3 种方式。①与文本有关的识别系统要求用户按照规定的内容发音, 并根据特定的发音内容建立精确的模型, 因而识别效果较好, 但系统需要用户配合, 如果用户的发音与规定的内容不符合, 则无法正确识别该用户。②与文本无关的识别系统则不规定说话人的发音内容, 因此建立精确的模型较为困难, 识别效果较差。无论与文本有关还是无关, 系统都无法区分是现场发音还是录音回放, 文本提示的识别系统可以有效防止这种情况的发生。③文本提示的识别系统随机产生发音材料, 并要求用户照此发音, 因此仿冒者无法事先知道发音内容, 但是该系统同样需要用户的配合。

说话人识别被认为是最自然的生物特征识别方式,语音的交互最易被人们接受。然而,若应用于大范围人群,语音并不能提供足够的信息来进行身份鉴别;同时,语音基本上属于一种行为特征,身体状况、紧张程度等因素会引起语音的改变。另外,环境噪声对语音识别的影响非常大。所以基于语音的身份鉴别系统的性能在很大程度上依赖于采集、传输、存储等数字化设备的精度,更重要的是该项技术需要说话人本身的配合。

从提高识别系统准确性和鲁棒性来说,说话人的语音识别与人脸识别、唇动识别等其他生物特征识别技术相结合(参见:多模态生物特征融合)是行之有效的办法。

参考文献

1. Campbell J P. Speaker Recognition: A Tutorial. Proc. IEEE, 1997, 85(9):1437~1462
2. Kersta L G. Voiceprint Identification. Nature Magazine, Dec. 29, 1962
3. Reynolds D A and Rose R C. Robust Text-Independent Speaker Identification Using Gaussian Mixture Speaker Models. IEEE Trans. Speech and Audio Processing, 1995, 3(1):72~83 (吴志勇 蔡莲红)

shuoming

说明 (declaration) 程序中用到实体的属性或定义描述。程序的说明部分使名字与被说明的实体相联系。例如,一个常量说明

```
PI:Constant:=3.1415926536;
```

使PI与数学中的 π 值相联系。

说明可分为常量说明,变量说明,类型说明,子程序说明,模块说明,异常说明等。**变量说明**定义了程序中使用的各种变量及其属性。类型说明除了通常的**类型定义**外,还有私有类型说明。当希望说明一个能为外界使用,但又要隐蔽其具体实现和内部数据结构的类型时,可以使用私有类型说明。**子程序说明**参见**函数与过程**。**模块说明**包括顺序模块说明与并发模块说明。**Ada语言**中的程序包为一种典型的顺序模块,它描述具有相对独立性的一组逻辑上相关的实体。例如,定义复数类型的数据结构及其操作(+, -, *, /, 求平方值)的程序包规约,在Ada中写为

```
package complex_arithmetic is
    type complex is
        record
```

```
        real-part:float:=0,0;
        imag-part:float:=0,0;
    end record;
    function "-"(a,b:complex)return complex;
    function "*" (a,b:complex)return complex;
    function "/"(a,b:complex)return complex;
    function sqr(a,b:complex)
end complex_arithmetic;
```

并发模块描述相对独立的可并行执行的一组逻辑实体。并发模块可在多机系统或多处理机系统上实现,也可在单处理机上以交叉执行的方式实现。例如,Ada中任务模块就是一种典型的并发模块,几个任务模块中各个任务除了发生会合的时刻外均为独立、并行地运行。异常说明是现代程序语言中,针对程序执行过程中可能发生的错误或异常情况作出处理的机制。这既是实时程序设计的需要,又可使程序的正常运行部分与出错处理(或异常处理)部分分离,提高了程序的清晰度与可理解性。

说明部分的主要任务是描述或定义各种实体,这里有一个作用域的概念,即每个说明的实体有这样的程序区域,在这个区域内可按定义给出的含义使用该名字。说明实体定义的作用域一般可静态确定,即由直接包含它的程序语法构造确定。与作用域相关的一个概念是作用范围,作用范围是指说明的标识符的可见范围。在分程序结构的语言中,内层作用范围可以不是外层作用范围的一部分。例如:以下Ada程序片段中,外部分程序A中内含一个内部分程序B,

```
A: declare
    alpha,beta:character;
begin
    get(alpha);
B: declare
    alpha:integer;
begin
    get(alpha);
    put(alpha);
end B;
put(alpha)
end A;
```

由于B中又说明了同名变量alpha,这样外部分程序A的alpha在B分程序中成为不可见的,因此,A的alpha的作用范围要去掉内部分程序B的这一区域。

程序语言的说明部分一般由编译程序在编译时处理完毕,但有的语言中(例如 Ada)的一些说明工作需要执行时才能处理。执行时处理说明部分的工作称为制作,它包括动态存储分配,计算初值表达式等工作。

参考文献

Ada Reference Manual ANSI/MIL-STD-1875A-1983
(陈涵生)

shuomingxing yuyan

说明性语言 (declarative language) 着重描述要处理什么,而非如何处理的非命令式语言。如何处理要取决于实现,后者包含直接反映在语言中的算法信息。说明性语言程序是关于问题解之约束陈述,这些约束迫使含于语言实现中的算法处理机制生成一个解或一组解。函数式(或作用式)语言与逻辑式语言均为说明性语言。

函数式语言的基本运算单位是函数,其程序是函数。函数从输入变元获取输入值,回送的结果即为函数本身之值。函数式语言的基本成分有:基本数据对象(如整数对象);定义新数据对象(如表的设施;基本函数(如基本算术运算);以及定义函数的机制。函数式语言是一种面向值的语言,无状态,因而无状态变化,无副作用。具有引用透明性,函数值只取决于变元值,具相同一组变元的函数,其值惟一。换言之,在程序中某处对一表达式求值并不影响其他表达式的求值,从而任一表达式的值由其变元值及所含常量惟一决定。

1960年诞生的 LISP 实际上已开创了函数式语言程序设计风格。1977年著名计算机科学家 J. Backus 在他接受 Turing 奖的演讲“程序设计能否从冯·诺依曼式的设计风格中解放出来?”中,从全新的角度深刻分析了冯·诺依曼计算机和以之为基础的命令式语言的本质缺陷,明确提出了一种纯函数式程序设计语言 FP 后,函数式语言发展迅速,例如,融进软件工程思想的 SML, ML, 商用纯函数式语言 Miranda 以及 Hope, Pebble, Haskell 等。

逻辑式语言的基本运算单位是谓词。谓词定义了变元间之逻辑关系。以逻辑式语言 Prolog 为例,它采用一阶谓词逻辑的子集——Horn 逻辑。其程序由围绕某一主题的事实、规则和询问三类语句组成,这三类语句分别用来陈述事实、定义规则和提出问题。程序的执行是根据预先读入之事实及其关系,按询问给出回答。在逻辑式语言程序中,程序与

数据统一,程序构造可以是增殖式的。

为在当前使用中仍占统治地位的冯·诺依曼体系计算机上寻求说明性语言的高效实现,不少说明性语言嵌入了若干命令式语言成分,从而多少掩盖了说明性语言的独特风格。

目前说明性语言尚处于蓬勃发展中,主要研究课题有:并行性、功效、大型软件编制以及面向对象语言的相互渗透与有机结合。

参考文献

1. Backus J. Can Programming be Liberated from the von Neumann Style? A Functional Style and its Algebra of Program. CACM. 1978, 21(8)

2. 郭浩志主编. 程序设计语言概论. 长沙:国防科学技术大学出版社, 1989 (郭浩志 徐家福)

siyao mima jishu

私钥密码技术 (private key cryptography)

在密码体制中加密和解密采用同一密钥的技术,又称对称密钥技术。从明文到密文的变换过程称为加密。变换本身是一个以加密密钥 k 为参数的函数,记作 $E_k(P)$ 。密文经过通信信道的传输到达目的地后需要还原成有意义的明文才能被通信接收方理解,将密文 C 还原为明文 P 的变换过程称为解密或脱密,该变换是以解密密钥 k' 为参数的函数,记作 $D_{k'}(C)$ 。在私钥密码系统中, $k = k'$ 并且 $D_{k'}(E_k(P)) = P$ 。

对称密码体制是从传统的简单换位、代替等密码技术发展而来的。自 1977 年美国颁布数据加密标准(DES)密码算法作为美国数据加密标准以来,对称密钥密码体制得到了迅猛发展,在世界各国得到了关注和使用。对称密钥密码体制从加密模式上可分为序列密码和分组密码两大类。

序列密码一直是作为军事和外交场合使用的主要密码技术之一。它的主要原理是,通过有限状态机产生性能优良的伪随机序列,使用该序列加密信息流,(逐位加密)得到密文序列。所以,序列密码算法的安全强度完全决定于它所产生的伪随机序列的好坏。

分组密码的工作方式是将明文分成固定长度的组(块),如 64 位一组,用同一密钥和算法对每一块加密,输出也是固定长度的密文。例如,数据加密标准(DES)密码算法的输入为 64 位明文,密钥长度为 56 位,密文长度为 64 位。

数据加密标准(DES)算法加密时把明文以位为

单位分成块,而后密钥把每一块明文转化成同样 64 位的密文块。DES 可提供 72 000 000 000 000 000 个密钥,用每微秒可进行一次 DES 加密的机器来破译密码需两千年。采用 DES 的一个著名的网络安全系统是 Kerberos,由美国麻省理工学院开发,是网络通信中身份认证的工业上的事实标准。

因为对称密钥密码系统具有加、解密速度快、安全强度高等优点,在军事、外交以及商业应用中使用越来越普遍。由于存在密钥发行与管理方面的不足,在提供数字签名、身份验证等方面需要公钥密码系统共同使用(参见公钥密码技术),可以达到更好的安全效果。

参考文献

1. 胡道元. 计算机网络(高级). 北京: 清华大学出版社, 1999

2. 卢开澄. 计算机密码学. 北京: 清华大学出版社, 1998 (胡道元)

sisuo

死锁 (deadlock) 多个进程因竞争共享资源而处于永远等待的状态。例如,一个系统含有资源 R_1, R_2 , 两个进程 A, B 分别占有其中一个资源而申请另一个资源,这样就出现了进程 A 占有资源 R_1 而等待进程 B 释放资源 R_2 ; 进程 B 占有资源 R_2 而等待进程 A 释放资源 R_1 , 于是这两个进程都不能执行下去而处于永远等待的死锁状态。

产生死锁的因素很多,不仅与系统拥有的资源数量有关,而且与资源分配策略,进程对资源的使用要求有关。出现死锁不仅会造成资源浪费,无法正常工作,严重时会使系统崩溃。因此,必须妥善解决和预防系统发生死锁。

有 3 种途径解决死锁问题:

(1) 死锁防止 是指采取措施保证系统不产生死锁。

系统产生死锁必须同时保持 4 个必要条件:

①互斥条件 进程互斥使用资源; ②部分分配条件 一个进程请求资源得不到满足而等待时,不释放已占有资源; ③不抢占条件 任一进程不能从另一进程那里抢夺资源; ④循环等待条件 存在一个循环等待链,其中,每个进程分别等待它前一个进程占有的资源。

只要能破坏上述 4 个必要条件的一个或几个,死锁就可防止。例如,系统采用资源静态分配法或资源层次分配法,则可破坏上述条件②或④,从而,

可防止系统发生死锁。

(2) 死锁避免 是指采取措施避免系统产生死锁。

防止死锁发生代价太高,使资源利用率低,系统效率差。如果系统动态分配资源,在分配过程中,掌握每个进程的资源申请情况,当把资源分配给申请者会产生死锁的话,就拒绝分配;否则接受申请,为进程分配资源。这样可动态测出发生死锁的可能性并加以避免。银行家算法是著名的死锁避免算法。

(3) 死锁检测 是指采取措施找出系统中已发生的死锁并加以解决。

对资源的分配加以限制不利于进程对资源的共享和提高系统效率。解决死锁的另一条途径是死锁检测,即对资源的分配不加任何限制,系统定时运行一个“死锁检测”程序,判断系统内是否发生死锁,若检测到死锁再设法加以解除。例如,可撤销某些有关进程,从一个进程那里剥夺资源等。进程资源图及 Petri 网等技术都可用来有效地判断系统是否发生死锁。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989 (郑宇华)

sushu

素数 (prime number) 除 1 和它自身外,不能被其他正整数整除的大于 1 的整数。例如 2, 3, 5, 7, 11, 13, 17 都是素数。除 1 以外不是素数的正整数称为复合数,1 是惟一的既非素数又非复合数的正整数。大约在公元前 300 年,欧几里得就证明了素数个数是无限的。目前所知道的最大素数是 $2^{859\,433} - 1$, 共有 258 716 位,是 1993 年发现的。

称 $M_p = 2^p - 1$ 形式的数为梅森数,其中 p 是素数。当 M_p 是素数时,称为梅森素数。目前已知道 32 个梅森素数,最大素数就是最大的梅森素数 $M_{859\,433}$, 最大的已知复合梅森数是 M_p , $p = 39\,051 \cdot 2^{6\,001} - 1$, 是 W. Keller 于 1987 年发现的。梅森素数是否有无穷多个,这是数论中尚未解决的问题。

称 $F_n = 2^{2^n} + 1$ 形式的数叫做费马数,当 F_n 是素数时,叫做费马素数。目前只知道 5 个费马素数 3, 5, 17, 257, 65 537。目前所知的最大复合费马数是 $F_{23\,471}$, 它有一个因子 $5 \cdot 2^{23\,473} + 1$, 是 W. Keller 于 1984 年发现的。是否有无穷多个费马素数,或者是

否有无穷多个费马数是复合数,都是没有解决的问题。高斯曾经证明过,如果 F_n 是素数,那么可以用圆规与直尺作出正 F_n 边形,这说明费马数与几何中某些问题有深刻的内在联系。

两个差等于 2 的一对素数叫做孪生素数,例如 3 和 5, 5 和 7, 11 和 13 等,目前所知道的最大孪生素数是 $107\,570\,463 \times 10^{2\,250} \pm 1$, 共有 2 259 位,是 Dubner 于 1985 年发现的。所谓孪生素数猜想是指存在无穷多对孪生素数。这个猜想至今没有解决,但认为它是正确的可能性很大。目前最好的结果是 1966 年我国数学家陈景润得到的: 存在无穷多个素数 p , 使得 $p+2$ 是不超过两个素数的乘积。

关于素数个数的研究是素数论中最重要的问题之一。以 $\pi(x)$ 表示不大于 x 的素数个数, 例如 $\pi(10)=4$, $\pi(100)=25$, $\pi(1\,000)=168$ 。前面说过的素数有无穷多个, 即 $\lim_{x \rightarrow \infty} \pi(x) = \infty$ 。此外容易证明 $\lim_{x \rightarrow \infty} \frac{\pi(x)}{x} = 0$, 这说明素数出现的概率为零。

所谓素数定理是 $\lim_{x \rightarrow \infty} \frac{\pi(x) \ln x}{x} = 1$, 最早是勒让德和高斯提出的猜想, 以后得到了证明。

如以 $\pi(x, q, l)$ 表示首项为 l , 公差为 q 的算术级数中不大于 x 的素数个数, P. G. Lejeune Dirichlet 于 1837 年证明了: 如果 l, q 是互素的正整数, 那么 $\lim_{x \rightarrow \infty} \pi(x, q, l) = \infty$ 。1949 年 A. Selberg 得到了它的初等证明。类似于素数定理, 对于固定的 q , 容易证明 $\lim_{x \rightarrow \infty} \frac{\pi(x, q, l) \varphi(q) \ln x}{x} = 1$, 其中 $\varphi(q)$ 是欧拉函数, 这就是算术级数中的素数定理。关于算术级数中的最小素数, 目前最好结果是陈景润 1979 年得到的。

参考文献

1. 华罗庚. 数论导引. 北京: 科学出版社, 1957
2. 华罗庚. 指数和的估计及其在数论中的应用. 北京: 科学出版社, 1963 (蒋增荣)

suxing ceshi

素性测试 (primality test) 数论中确定整数 N 是素数还是复合数的方法。在许多情况, 例如 RSA 公钥密码系统以及各种建立在有限域上离散对数问题的密码系统, 都需要知道一个大整数是否是素数。近十几年来, 这个问题发展极为迅速, 目前已取得了突破, 在 CDC CYBER 170-750 计算机上, 大约 30 秒就可确定 100 位大整数的素性, 约 8 分钟可确定 200

位大整数的素性。最基本、最简单的素性测试是试除法, 设 N 是奇整数, 用一个奇整数 m 去除 N , 若 m 整除 N , N 是复合数, 否则, 就称 N 通过了“ m 试除”的素性测试。当 N 通过更多的试除测试时, N 愈来愈可能是素数, 当 m 穷尽 3 到 \sqrt{N} 的一切奇整数时, N 就确实是素数了。试除法不仅能测试 N 的素性, 还能分解出 N 的因子。但当 N 没有小因子时, 试除法的运算时间太长, 其时间复杂性是指数级的, 在每秒百万次的计算机上, 当 $N \approx 10^{40}$ 时, 算法时间差不多要 100 万年。

概率素性测试法是一种运算时间非常长 (也许是无穷长) 的测试法, 但实际执行时, 都有一个特殊的指令, 当超过一定时间或一定步数后, 算法就将停止, 但是如果未达到规定的时间或步数算法便告结束, 结果就是“真的”, 即 N 确实是一个复合数。其一般形式是“对于输入奇整数 N , 在 $[1, N-1]$ 中随机选取整数 b 作为基, 对 N 关于 b 执行某种测试, 一旦 N 在测试中失败, 算法就停止。”当 N 是素数时, 测试将无限进行下去, 除非得到停止的指令。最有实际应用价值的是 Miller-Rabin 概率测试法, 它是进行如下的测试: 设 N 为奇整数, 记 $N-1=2^t t$ (t 为奇整数, $s>0$), 若存在正整数 $b \in [1, N-1]$, 满足 $b^t \equiv 1 \pmod{N}$, 或 $b^{2^r t} \equiv -1 \pmod{N}$ ($0 \leq r \leq s-1$), 就称 N 关于 b 通过测试, 否则测试失败。由于任何素数都能通过测试, 一旦对 N 测试失败, N 必为复合数。如果 N 通过测试, N 为复合数的概率至多是 $1/4^k$, 通过 k 次, 为复合数的概率不超过 $1/4^k$ 。 k 愈大, N 愈可能是素数。Rabin 对 $(2^{400}-593)$ 进行了 $k=100$ 次测试均获通过后才断言它是一个素数, 这个数是小于 2^{400} 的最大素数。该方法的时间复杂度为 $O(\ln^3 N)$ 。其他还有 Solovay-Strassen 概率测试法, 它的时间复杂度也是 $O(\ln^3 N)$ 。 N 通过这种测试仍为复合数的概率不超过 $1/2$ 。确定型素性测试法是一种能证明 N 确实是素数的测试法。这类严格的素性证明必须完全排除其“伪”素数的可能, 目前的算法大都或者只有理论意义或者非常复杂。例如根据 Wilson 定理, 只要 N 整除 $(N-1)!+1$, N 就是素数, 但它看来没有应用价值, 因为当 N 是一个 3 位数时, $(N-1)!+1$ 就超过 100 位, 计算量十分巨大。1876 年, E. Lucas 给出如下确定型测试法: 如果同余式 $g^x \equiv 1 \pmod{N}$ 仅对于 $x=N-1$ 成立而对于 N 的其他真因子不成立, 那么 N 是素数。为了求得 g , 首先必须将 $N-1$ 进行分解, 当 N 很大时, 这是一件很困难的工作, 目前最快方法的运算时间也是亚指数级的。

$O(\exp(c\sqrt{\ln N \ln \ln N}))$ 。对于 $N-1$ 有小因子的整数 N , Lucas 测试法特别有效, 特别可以得到有关费马数 $F_n = 2^{2^n} + 1$ 的 Pepin 测试法: 当 F_n 整除 $(3^{(F_n-1)/2} + 1)$ 时, F_n 是素数。同样, 如能分解 $N+1$, 也易于证明 N 的素性。Lucas 和 D. H. Lehmer 利用这点得到著名的有关梅森数 $M_p = 2^p - 1$ 的测试法: 设 p 是素数, 当且仅当 $v_{p-2} \equiv 0 \pmod{M_p}$ 时, M_p 是素数, 其中 $v_0 = 0, v_s = v_{s-1}^2 - 2 (s = 1, 2, \dots, p-2)$ 。应用这个测试法和其他简单的测试法, Lucas 在 1976 年证明了 M_{127} 是素数, 这是使用计算机前找到的最大素数。1975 年, Lehmer 等人指出, 只要知道 $N \pm 1, N^2 \pm N + 1, N^2 + 1$ 的部分分解, 就能证明 N 的素性, 这些方法的最大成功是证明了 $2^{400} - 593$ 是素数。1980 年一些学者利用“雅可比和”的概念提出了时间复杂度差不多是多项式 $O((\ln N)^{\ln \ln N})$ 的测试法。此外还有椭圆曲线测试法。目前已经证明, 如果广义黎曼假设 (ERH) 成立, Miller-Rabin 概率测试法的时间复杂度是多项式 $O(\ln^5 N)$ 的确定型测试法, 只要对 N 关于小于 $\ln^2 N$ 的 b 进行 Miller 测试, 如果均获通过, N 就是素数。

参考文献

1. Koblitz A. A Course in Number Theory and Cryptography. New York: Springer-Verlag, 1987
2. Riesel H. Prime Numbers and Computer Methods for Factorization. Boston: Birkhäuser, 1985

(蒋增荣)

suanfa

算法 (algorithm) 解题过程的精确描述, 由有限条可完全机械执行的、有确定结果的指令 (或命令、语句) 构成。关于问题求解, 一方面是问题的描述, 另一方面是用于求解此问题的装置。关于问题描述的一般要求是: 题意准确、清晰、明了, 解的规格确定。至于解题装置, 可以是机器, 也可以是人, 也可以是两者的结合。算法是用计算装置能够理解的语言描述的解题过程, 具有如下性质:

(1) 将它作用于所求解的问题的给定输入集上, 或作用于问题自身的描述上, 将产生惟一确定的动作序列, 此序列是有限的;

(2) 此序列或终止于给出问题的解或终止于指出问题对此输入数据无解。

下面用一个例子来解释上述概念。问题描述是: “求实数 x 的平方根”。该问题的题意明确, 但

没有指明解的规格。如果满足于把“ \sqrt{x} ”看成是问题的解, 那么, 解决此问题的算法再简单不过了: “将根号 $\sqrt{\quad}$ 与输入数据 x 相连接”。如果要求问题的解是精确的十进制表示, 那么, 2 的平方根就永远也算不出来。因为只用“有限”个动作是无法完成的。

如果把问题的描述改为: “求实数 x 的正平方根, 准确到十进制表示的小数点后四位”。这个描述有许多改进: ①它指明要求最终解是正根, 而先前的那个描述对解的要求是不确定的; ②它排除了把“ \sqrt{x} ”作为问题解的可能性; ③“小数点后四位”提供了测试最终解的规格标准。

对于上述精确描述了的问题的解法可粗糙地表示为:

(1) 选择一个数 y , 计算其 y^2 。

(2) 若 $|y^2 - x| < 5 \times 10^{-5}$, y 即是所要的解, 否则返回到步骤 (1)。

这个方法不是上述意义下的“算法”。因为选择 y 的初值的初始动作是不确定的, 而且每个动作的后续动作 (即如为何选择 y 的下一个值) 也是不确定的, 因此, 将上述的过程作用于输入值 x 后无法得到“惟一确定的一串动作序列”。

若将上述的方法改进为:

(1) 选择 $y = 1$ 。

(2) 计算 y^2 。

(3) 若 $|y^2 - x| < 5 \times 10^{-5}$, y 即是所要的解, 停止; 否则, 转到步骤 (4)。

(4) 用 $((x/y) + y)/2$ 作为下步的 y 值; 转到步骤 (2)。这个过程是牛顿-拉弗森方法的特例。它消除了上述粗糙方法的弊端。并且可以证明, 将它作用于任何非负实数 x , 在有限步之内一定会得到满足解的规格的正平方根。因此, 这个过程已经非常接近于前述意义下的“算法”。只是, 若将这个解法作用于负实数 x , 解题过程就会无休止地计算一个又一个的 y , 永不终结。即动作序列的有限性不能保证 (这种不终结的过程称为半算法)。因此, 需要对上述方法再作点改进, 在它的前面加一步:

(0) 若 $x < 0$, 则无解, 停止; 否则, 转到步骤 (1)。

这样就完美了, 称得上是一个完全的“算法”。

上述算法是用自然语言描述的, 易于为人所理解。不难用计算机所能接受的 PASCAL 语言将其改

写成如下的程序：

```

program Square Root(input, output);
var X,Y,Z: real;
begin
  read ln(X);
  write ln('Find the square root of', X: 0:
0);
  if x ≥ 0 then
    begin
      Y: = 1; Z: = Y * Y;
      while abs(X - Z) ≥ 0.00 005 do
        begin
          Y: = ((X / Y) + Y) / 2;
          Z: = Y * Y
        end;
      write ln('The square root of', X: 7:
5, 'is', Y: 7: 5, ' . ')
    end
  else
    write ln('There is no square root for',
X: 8: 5, ' . ')
end

```

这个程序执行时将输入所要求根的数据,然后给出回答:或给出平方根,或指出无解。这个程序对输入数据 85, 1, 0. 61 825, -9 的相继执行结果是:

```

Find the square root of 85
The square root of 85.00 000 is 9. 21 954.
Find the square root of 1
The square root of 1.00 000 is 1. 00 000.
Find the square root of 0. 61 825
The square root of 0. 61 825 is 0. 78 628.
Find the square root of -9
There is no square root for -9.

```

在上述关于“算法”的严格定义中,我们坚持算法作用于给定的输入数据后所产生的动作序列的惟一性与有限性。惟一性蕴涵着确定性,有限性蕴涵着终止性。确定性指初始动作是确定的,每个动作的后续动作也是确定的(“停止”动作无后续动作)。如果准许每个动作的后续动作可以有有限多种选择,即下一动作是不确定的,这种算法称为不确定算法。例如走迷宫,从其入口到出口,用不确定算法描述起来就比较简单。

设计算法与分析算法是计算机科学的核心问题。从应用范围来看,算法可分成数值算法和非数值算法两大类。从工作方式来看,算法可分成串行算法和并行算法两大类。早在 20 世纪 70 年代, D. E. Knuth 就指出,计算机科学就是研究算法的学问。

因此,将算法这个概念严格化、精确化有重要的实际意义。

任何计算机程序至少是一个半算法,凡能达到“停止”的程序是一个算法(当然未必能解决程序人员心目中的问题)。对于一个给定的问题,可能有许多个算法(程序),但它们的质量不会全同。衡量程序质量的主要因素是,程序执行所费时间之长短和所需空间(存储容量)之多少。就当今的计算机硬件而言,尤以时间因素为贵。另一个重要的质量因素是算法的一般性问题(即适用范围之大小问题),如何裁剪一般性主要取决于应用背景。当然还有其他的质量因素,例如,对于数值算法而言,有收敛速度之高低,误差积累之快慢,结果精度之好坏,等等。关于算法的时、空效率参见**计算复杂性理论**。简言之,将算法概念严格化,既有利于算法设计又有利于算法质量分析。

把算法概念精确化还有深刻的理论意义。原先人们认为,任何合式描述的数学问题都是可计算的。20 世纪 20 年代开始,数学家对此提出了疑问:问题的“可解性”或函数的“可计算性”到底意味着什么?为了回答这个问题,几十年来建立了许多理论,如图灵机理论、递归函数理论、正规算法理论、λ-演算等等,并证明了所有这些理论是等价的:即,一个问题在一个概念下可解,在所有别的概念下也可解。这些研究启示了,算法过程是机械的,发明算法才是人类的创造。

参考文献

Aho A V, Hopcroft J E and Ullman J D. The Design and Analysis of Computer Algorithms. Reading: Addison - Wesley, 1974 (朱洪 陈火旺)

suanfa donghua

算法动画(algorithmic animation) 采用算法来控制动画形体的运动和变化的**计算机动画**技术。又称过程动画。在这种动画中,运动由变换表(旋转、位移、缩放、切变、扭曲、弯曲、随机变换、颜色改变等)或带有参数的过程来描述,每个变换由一些参数定义,这些参数可按照一定规律改变,这些规律可以使用解析形式给出或者使用复杂过程(如微分方程组)的解来定义。

最简单的算法动画是按照物理定律(包括运动学、逆运动学、动力学和逆动力学等),用一个数学模型去控制物体的几何形状和运动,如水波随风的运动,物体的变形、弹跳、碰撞运动等。另一类算法

动画为粒子和群体的动画。粒子系统是最实用的算法动画技术之一,其基本思想是将许多简单形状的微小粒子作为基本元素聚集起来形成一个不规则的模糊物体,从而构成一个封闭的系统。粒子具有生命周期,它们经历出生、成长、衰老和死亡的过程。粒子系统可成功模拟一些自然景象,如火、烟、喷泉等。

参考文献

Thalmann N M, Thalmann D. Computer Animation Theory and Practice. Second Revised Edition. Tokyo, Berlin, Heidelberg, New York: SpringerVerlag, 1990
(王裕国 金小刚)

suanfa sheji

算法设计 (design of algorithm) 研究算法设计一般方法的学科。算法设计是算法求解问题时首先要考虑的问题,一旦设计出算法,除分析算法复杂性等性能外,还要讨论能否设计出更好的算法。下面介绍一些设计算法的基本策略。

分治法 将一个问题分解成若干个子问题,原问题的解由其子问题的解通过某种方法组合而成,每个子问题又分成若干个更小的子问题,子问题的解由更小的子问题的解用同样方法组合而成,……,这样一直分下去,直到子问题的规模足够地小,以至于能有简单算法直接解决。一般地,当子问题的规模大致相等时分治效率最高。分治法的典型例子是合并排序;将要排序的数据分成个数几乎相等的两个子集,先对这两个子集排序,然后将这两个排好序的子序列合并成一个有序序列。至于如何对这两个子序列排序,则是用递归的方法,把子集分成更小的子集,将其排序,然后合并而得。快速排序,快速傅里叶变换,Strassen 的矩阵乘法等都是采用分治法的典型例子。

贪心法 在给定输入集和判别一个解是否最优的准则之后,贪心法主张算法的每一步都要使其解不但可行而且比前一步更优,这样不断优化下去,直到最优解获得为止。求图的最小生成树的 Kruskal 算法是贪心法典型实例;它先把边按权值递增排序,依次取出边,加入到初始时空集的生成树的边的集合中,如果加入的边使生成树的边集产生回路就不加入,这样逐次将剩下的权值最小的边依次加入或不加入(取决于是否产生回路)直到生成树边集的边邻接图中所有点为止。可以证明,上述算法最后必输出最小生成树。一般来说,贪心法是否能输

出最优解,是需要证明的。成功的贪心法例子还有求指定点对时间的最短路径的 Dijkstra 算法等。虽然贪心法并不一定总能产生最优解,但有时能帮助我们找到近似解。

动态规划 一种多阶段决策策略。它的依据是最优化原理:作为整个过程的最优决策序列具有下述性质,不论过去的状态和决策如何,当前的决策必须对前面的决策所形成的状态构成最优决策序列。根据这个原理,在决策过程的第 i 步,不必考虑前 $i-1$ 步的决策,而只需要考虑在这一步形成的状态,过去的历史只能通过当前的状态去影响未来,当前的状态就是未来过程的初始状态。求最短路的 Floyd 算法,二叉最优搜索树的 Huffman 算法,矩阵列相乘的最优安排均为动态规划算法。

深度优先和广度优先搜索 这是有关图的算法的常用途径。算法沿着图的每个点和它的某一邻边进行搜索(通常要根据问题的自身要求做一些工作),如果该边的另一端点已访问过,则放弃,否则转入该边的另一端点,从新的端点出发,沿着其邻边再搜索到该邻边的另一端点,依此类推,直到所有点被搜索到为止,这样的搜索称为深度优先搜索。如果不是马上转入邻边的另一端搜索下去,而是先把某一点的各邻边都搜索一遍(也要根据问题的自身要求做一些工作),然后再向某一邻边的另一端点(如果它还未访问过)搜索下去,这样的搜索法叫广度搜索法。求图的连通分支,双连通分支,强连通分支,前、中、后序遍历等都采用了广度或深度优先搜索法。对某些 NP 难问题,例如旅行商问题,为了避免搜索全部结点和边(可能有指数多个!),设置目标或价格函数,每次选当前目标或价格值最优的结点进行搜索,剪除那些达不到(或看似达不到)最优的结点,不再对它及其子孙结点搜索下去。这种算法称为分枝限界法,它有时能产生近似较优解。人工智能中的“and/or”图的 A^* 搜索法也基于和分枝限界法相同的思路。

回溯法 如果所期待的解表示为形式 (x_1, x_2, \dots, x_n) , x_i 选自某有限集 S_i , 回溯法是适当和算法设计策略。这本质上是穷举法,它的基本思想是:通过逐步扩张部分解 (x_1, x_2, \dots, x_i) , $i = 1, 2, \dots, n$ 进行搜索,部分解满足一定的约束条件使之成为可行解,我们可以想象以部分解为结点,为一种状态空间,从而构造状态空间树,回溯法是一种对状态空间树的深度优先搜索,若部分解 (x_1, x_2, \dots, x_i) 满足约束条件,可以扩充为可行解,则添加 $x_{i+1} \in S_{i+1}$, 即搜

索到结点 $(x_1, x_2, \dots, x_i, x_{i+1})$ 是否满足部分解的约束条件, 满足了则再向下搜索, 即继续添加 $x_{i+2} \in S_{i+2}$ 。若所有的 $x_{i+1} \in S_{i+1}$ 都不能得到满足约束条件的部分解, 就去掉 x_i , 回溯到 $(x_1, x_2, \dots, x_{i-1})$, 添加那些尚未考查过的 $x_i \in S_i$, 看其是否满足约束条件, 如此反复进行, 直到得到可行解(它位于状态空间树的叶结点)或证明无解(当状态空间树全部搜索到而不能满足约束条件时)。八皇后问题, 求哈密顿回路问题都可用回溯法解答。

归约法 将一个问题 II 的实例多项式时间内转换到另一问题 II' 的一个实例, 使得原实例有解当且仅当新问题 II' 的实例有解, 而且两个问题的解有一定的联系。这种归约原来只对判定问题进行, 现在已扩大到优化问题, 扩大到优化问题的近似解法, 或者用以判定满足预定近似比的近似较优解存在还是不存在(如果 $P \neq NP$, 参见 **NP 完全性理论**)。

如果我们知道一个算法将执行多次, 就可以先作一些准备性计算, 以便在施行以后的算法步时大大加快速度, 这种方法被称作预算法, 它可用于计算同一多项式在若干点上的值, 求某一模式在不同正文中的串匹配等。如果运用到集合的合并和寻找属员问题上, 则是有名的路径压缩法。预算法被系统地发展成时空折算和积存摊还算法, 与之相应, 还设计出为路径压缩、积存摊还算法服务的数据结构。

参考文献

1. Aho AV, Hopcroft J E, Ullman J D. The Design and Analysis of Computer Algorithms. Addison-Wesley Publishing, 1974
2. Cormen T H, Leiserson C E, Rivest R L. Introduction to Algorithms. The MIT Press, 1990
3. 朱洪, 陈增武, 段振华, 周克成. 算法设计和分析. 上海: 上海科学技术文献出版社, 1989

(朱洪)

suanfaxue

算法学 (algorithmics) 系统地研究算法的设计、分析和验证的学科。算法是一个很古老的数学概念。最基本的算法是加、减、乘、除。它们是小学生学习算术的入门方法。现代计算机出现以来, 人们不断用计算机求解一些空前大型和复杂的数学问题。但是, 凡与计算机打交道, 无不是研究设计各种类型的算法(用某种计算机语言表示, 如 PASCAL,

并由计算机执行)。因此, 将算法本身作为研究对象, 建立一门独立的学科已是水到渠成。这就是算法学的基础。算法学并不包罗万象地研究设计各类数学问题的解的具体算法(这是各类具体计算学科的任务, 如计算数学、计算力学、计算物理、计算化学等), 而是从方法学角度, 研究算法的设计、描述、确认和品质评审。算法学是计算机科学最重要的内容。有的计算机学者甚至称, 计算机科学就是算法的科学。

算法学的主要内容大体可分为: 设计、验证和分析。“设计”指创作算法的过程和研究有代表性的、好的创作策略(例如自顶向下策略, 分而治之策略)。“验证”是指证明算法的正确性, 即证明它满足所欲求解的问题的要求。最基本的证明手段之一是数学归纳法, 通常用于证明循环不变式之成立。“分析”是指对算法的效率的确定。如算法所需执行时间之长短和运行空间之多少。时空效率通常表示为输入数据量的多少的函数。如果同一问题存在几个不同的算法解, 分析的任务之一就是比较这些算法效率的高低。通常所说的算法复杂性就是指算法效率的高低。效率越低的算法意味其复杂性越高; 效率越高的算法意味其复杂性越低。分析任务中最难的工作是研究问题复杂性(亦称计算复杂性), 即确定一个问题的所有可能的算法解的最优效率或者最优效率的界限。

作为一个简单例子, 考虑多项式的计算问题:

$$p(x) = a_n x^n + \dots + a_1 x + a_0$$

乍看起来, 多项式计算不成问题, 对任何 x , 不难算出 $p(x)$ 的值。但若 n 非常大, 或者我们面对着许多个 x 的值而必须计算许多个 $p(x)$ 的值时(这种情形可能来自编码理论或多项式逼近), 这时就应该考虑设计更能胜任的计算方法了。

例如, 按上述的多项式表示形式, 采用最直接的算法, 自左至右逐项孤立地计算, 算 $a_n x^n$ 需作 n 次乘法, $a_{n-1} x^{n-1}$ 需作 $n-1$ 次乘法, 如此等等, 计算整个 $p(x)$ 需作 $n(n+1)/2$ 次乘法和 n 次加法。这样的算法, 效率是很低的。试想一下, 如果 x 的幂次不重复计算, 那么乘法的次数将下降为 $2n-1$ 次。如果还坚持从左算到右, 即从 $a_n x^n$ 算起, 那就需要把 x 的幂次的中间结果保留下来, x^n, x^{n-1}, \dots, x^2 需要 $n-1$ 个单元。构造这一串中间值只需要 $n-1$ 次乘法。这样算法的效率为 $2n-1$ 次乘法, n 次加法和 $n-1$ 个中间结果单元。但若从多项式的右端算到左边, 即从 $a_0 + a_1 x$ 算起, 那么 x 的幂次

只需保留当前值即可,即仅需一个中间单元。这样算法的效率就更高一点,即为 $2n-1$ 次乘法, n 次加法和—个中间单元。这个算法可用类 PASCAL 语言表示如下:

基本算法

```
Poly ← a0 + a1 * x;
Power ← x;
for k = 2 to n do
  Power ← Power * x;
  Poly ← Poly + ak * Power
end for;
output( Poly )
```

我们还可以把计算多项式的算法进一步优化,只需把 $p(x)$ 改写成如下的嵌套形式:

$$p(x) = (\cdots (a_n x + a_{n-1})x + \cdots)x + a_0$$

按照这种形式,采用嵌套乘、加算法, $p(x)$ 的计算仅需 n 次乘法和 n 次加法。这个算法可用类 PASCAL 语言表示为:

嵌套算法

```
Poly ← an;
for i = 1 to n do
  Poly ← Poly * x + an-i
end for;
output( Poly )
```

这个算法的效率比前述的基本算法几乎要高出一倍。

嵌套算法有点不太直观,何以证明它是有效的,即它确实计算了多项式 $p(x)$ 呢? 这可以通过检查循环不变式而得到证实。可用归纳法证明: 在通过第 i 次循环之后, $\text{Poly} = \sum_{j=0}^i a_{n-j} x^{i-j}$ 成立(这个式子即所谓循环不变式)。当整个循环结束时, $i = n$,

$$\text{Poly} = \sum_{j=0}^n a_{n-j} x^{n-j}, \text{ 即 } \text{Poly} = p(x)。$$

实际上,还可以证明嵌套算法是计算多项式的最优算法。嵌套计算方法是一种很老的算法。但证明它的效率最优性却是相当后来的事情。

通过上述例子,讨论了计算多项式 $p(x)$ 的算法,验证了它们,并分析比较了它们的效率,最后指出嵌套算法的最优性。这就是算法学所关心的主要内容。多项式的计算虽然不难,但上述的展开过程使我们看到研究算法学意义之所在。在算法学的领域中,难题比比皆是,许多新、老问题有待探索和研究解决。

参考文献

Sara BAASE 著. 计算机算法; 设计和分析引论. 朱洪等译. 上海: 复旦大学出版社, 1985

(陈火旺 贵可荣)

suanshu luoji bujian

算术逻辑部件 (arithmetic and logic unit, ALU)

对以机器字表示的操作数进行算术和逻辑运算的功能部件。它是数字计算机运算器的核心部件。ALU 通常由加法器和功能控制电路组成。在有些计算机系统中, 算术运算和逻辑运算分别用算术部件 (AU) 和逻辑部件实现。很多计算机不止一个 AU, 而且 AU 常分成定点 AU 和浮点 AU, 多处理机系统可包含多个相同的 ALU。

在数字计算机发展的历史中, 美国 Intel 公司的 SN 74181 中规模集成电路是很流行的 4 位 ALU, 该器件框图如图 1 所示。

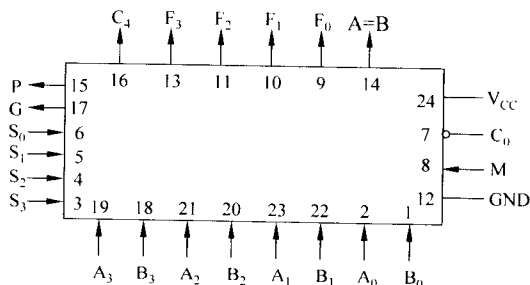


图 1 SN 74181 框图

图中的数字是引脚号, 各引脚的功能如下:

(1) $A_0 \sim A_3$ 和 $B_0 \sim B_3$ 分别是两个 4 位数的输入。

(2) M 为控制位, 高电平时, SN74181 进行逻辑运算; 低电平时, 进行算术运算。

(3) $S_0 \sim S_3$ 是功能控制引脚, 不同的组合 $S_3 S_2 S_1 S_0$, 可得到不同的功能, 逻辑运算和算术运算各有 16 种。

(4) C_0 为低位进上来的进位信号。

(5) $F_3 \sim F_0$ 为算术逻辑运算的结果输出。

(6) C_4 为向高位的进位信号。

(7) 14 脚输出高电平, 表示 $A = B$ 。

(8) P 和 G 是与并行进位链电路连接的输出信号, 其中 G 为进位产生输出, P 称为进位传递输出。

SN 74181 的正逻辑功能表如表 1 所示。在功能表中, “加”表示算术加, “+”表示逻辑加; “减”

表示算术减,“-”表示逻辑减。

表1 SN 74181 正逻辑功能表

功能选择	M = H	M = L	算术运算
$S_3 S_2 S_1 S_0$	逻辑运算	$C_0 = 1$	$C_0 = 0$
L L L L	\bar{A}	A	A 加 1
L L L H	$\bar{A} + \bar{B}$	A + B	(A + B) 加 1
L L H L	$\bar{A} \cdot \bar{B}$	A + \bar{B}	(A + \bar{B}) 加 1
L L H H	“0”	减 1	“0”
L H L L	$\bar{A} \cdot B$	A 加 ($\bar{A} \cdot \bar{B}$)	A 加 ($\bar{A} \cdot \bar{B}$) 加 1
L H L H	\bar{B}	($\bar{A} \cdot \bar{B}$) 加 (A + B)	($\bar{A} \cdot \bar{B}$) 加 (A + B) 加 1
L H H L	$\bar{A} \oplus \bar{B}$	A 减 B 减 1	A 减 B
L H H H	$\bar{A} \cdot \bar{B}$	$\bar{A} \cdot \bar{B}$ 减 1	$\bar{A} \cdot \bar{B}$
H L L L	$\bar{A} + \bar{B}$	A 加 ($\bar{A} \cdot \bar{B}$)	A 加 ($\bar{A} \cdot \bar{B}$) 加 1
H L L H	$\bar{A} \oplus \bar{B}$	A 加 B	A 加 B 加 1
H L H L	\bar{B}	($\bar{A} \cdot \bar{B}$) 加 (A + B)	($\bar{A} \cdot \bar{B}$) 加 (A + B) 加 1
H L H H	$\bar{A} \cdot \bar{B}$	($\bar{A} \cdot \bar{B}$) 减 1	$\bar{A} \cdot \bar{B}$
H H L L	“1”	A 加 A	A 加 A 加 1
H H L H	$\bar{A} + \bar{B}$	A 加 (A + B)	A 加 (A + B) 加 1
H H H L	A + B	A 加 (A + \bar{B})	A 加 (A + \bar{B}) 加 1
H H H H	A	A 减 1	A

SN 74181 是由非门、与或非门、与非门和异或门构成的组合逻辑电路,其基本逻辑结构是超前进位加法器。如果把 SN 74181 电路中的并行进位链省略掉,则其 1 位的逻辑电路图如图 2 所示。

从图 2 可以看出,三与或非门受 S_0 和 S_1 控制,二与或非门受 S_2 和 S_3 控制。 $S_0 S_1$ 与 X_i 的关系及

$S_2 S_3$ 与 Y_i 的关系如表 2 所示。

表2 X_i 和 Y_i 的值

$S_1 S_0$	X_i	$S_3 S_2$	Y_i
0 0	\bar{A}_i	0 0	1
0 1	$\bar{A}_i \cdot \bar{B}_i$	0 1	$\bar{A}_i + B_i$
1 0	$\bar{A}_i \cdot B_i$	1 0	$\bar{A}_i + \bar{B}_i$
1 1	0	1 1	\bar{A}_i

当控制端 M 为高电平时, $F_i = X_i \oplus Y_i$, 根据 $S_3 S_2 S_1 S_0$ 的不同取值, X_i 和 Y_i 就有 16 种组合,即得到表 1 的 16 种逻辑运算功能;当 M 为低电平时, $F_i = X_i \oplus Y_i \oplus C_i$, 也就是 $F_i = \bar{X}_i \oplus \bar{Y}_i \oplus \bar{C}_i$, 即 F_i 为本位和的反码,这种情况下, X_i 和 Y_i 同样也有 16 种组合,实现 16 种算术运算功能。

采用 4 片 SN 74181, 把它们的进位信号串接在一起,即可组成一个 16 位组间串行进位的 ALU。若用 4 片 SN 74181, 再加上一片超前进位链(CLA)集成电路 SN 74182, 即可组成一个 16 位组间并行进位的 ALU。

随着器件集成度的提高,可将更多位的 ALU 集成在一个芯片内。例如,20 世纪 80 年代后期 AMD 公司的 AM 29332 就是一个 32 位 ALU 器件。但是在微处理器出现以后,ALU 就不再作为独立的电路出现,而只是微处理器芯片的一部分。例如,8086 微处理器中,ALU 和运算寄存器、标志寄存器及通用寄存器一起构成 8086 的执行部件。

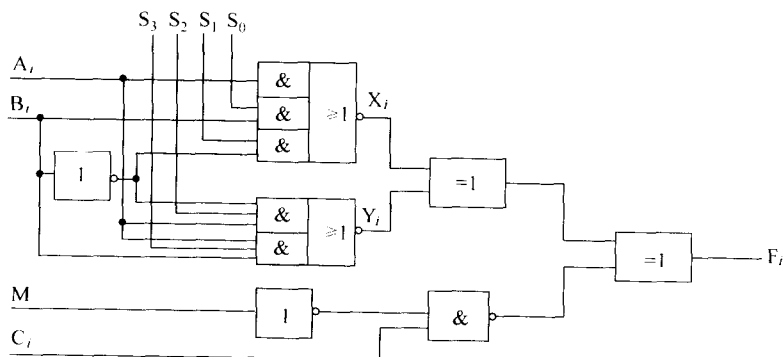


图2 SN 74181 的 1 位逻辑电路图

对于 80386, ALU 只是执行定点加、减类运算和逻辑运算的部件,乘除部件独立出来,而浮点运算则

靠 80387 完成。到了 Pentium(奔腾)处理器,有两个整数 ALU,而浮点部件又由加法单元、乘法单元和

除法单元组成。

尽管器件不同,但这些微处理器的 ALU 的基本原理还是相似的。

参考文献

1. 李文兵. 计算机硬件原理教程. 天津:天津科技翻译出版公司,1994
2. 王爱英主编. 计算机组成与结构(第三版). 北京:清华大学出版社,2001 (刘恩德)

suanshu luoji yunsuan

算术逻辑运算 (arithmetic /logic operation) 计算机对数据所进行的算术运算和逻辑运算。

随着计算机应用范围的不断扩大,计算机能表达、传送、存储和处理的数据有数字、字母、符号、图形、图像和声音等多种形式。尽管数据的表达形式、处理方法以及解决问题的算法有很大差别,但最终可以分解为一组或若干组遵循一定次序关系的算术运算和逻辑运算。因此算术运算、逻辑运算以及控制这些运算的机制构成了计算机运行的基础。

人们习惯于用十进制记数法来表示数的数值,但在计算机内部运算时通常采用二进制记数法,这是因为二进制数的每一位只有“1”或“0”两个值,它们可用电子线路(如触发器)的两种状态来表示。二进制运算规则比较简单,也便于在计算机中实现。一般人机交互的数据是以十进制形式表示的,而在计算机内部则经过转换后变为二进制形式。

计算机的基本算术运算有加法、减法、乘法和除法运算,为了便于实现加减法运算,除了一般的表示方法(原码)以外,又采用了补码和反码来表示数(参见数制)。为了使数有不同的精度和范围,又有定点数和浮点数两种表示方式,并制定了浮点数标准。至于对二进制数如何进行加、减、乘、除运算可参见二进制算术运算。

一个二进制码只有“1”和“0”两个值,如果赋以逻辑属性,例如表示成“真”和“假”,或者表示成“是”和“非”,将这种二进制码表示的变量称为逻辑变量,描述逻辑变量关系的函数称为逻辑函数,对逻辑变量和逻辑函数进行描述和运算的数学工具称为逻辑代数,也称为布尔代数,实现逻辑功能的电路称为逻辑电路。逻辑电路除了用于计算机的算术逻辑部件中以外,在计算机的其他部件中也广泛应用。有关基本逻辑运算的规则及逻辑电路的表示方法可参见逻辑运算。(王爱英)

suiji cunqu cunchuqi xinpian

随机存取存储器芯片 (random access memory chip) 可随机地对给定地址的存储单元进行写入和读出操作的半导体存储器芯片。存储在存储单元的数据在断电后可能会丢失。

随机存取存储器(RAM)芯片由存储单元阵列和外围电路集成。前者存储数据,后者对给定地址所选择的存储单元进行读写操作的控制。外围电路包括地址缓冲和地址译码驱动电路、读出放大电路和输出数据缓冲电路、写入数据缓冲和驱动电路以及片选和内部时序控制电路等。

根据数据存储方式,RAM 芯片可分为静态随机存取存储器(SRAM)芯片、动态随机存取存储器(DRAM)芯片和伪静态 RAM 芯片(或单管 SRAM)三种。

静态随机存取存储器芯片以两个反相器交叉耦合所构成的触发器存储数据,再由两个晶体管作为选择和读写控制,形成选择和读写控制存储单元。有电源时,由于触发器的双稳定状态,所存储的数据不会丢失,不需定时刷新。列线的读出写入信号幅度大,存取速度快,接口控制电路简单。但存储单元占用芯片的面积大,每位平均价格高。SRAM 芯片主要用于组成计算机高速缓冲存储器,以及存储容量小、需要简单接口控制的数据存储器。

动态随机存取存储器芯片采用一个小电容和一个 MOS 开关作为存储单元。电容存储数据;MOS 管用作行选择,使选中行的单元可通过列线输出电容的状态(读出)或改变电容的状态(写入)或刷新。由于漏电,电容所存的电荷会有变化,产生读出数据的错误。因而所存储的数据必须定时刷新以免消失。此外每次行选时,选中行存储单元的电容上的电荷与所连列线分布电容重新分配,将所存数据读出。但由于区分为 1 或为 0 数据的列线电平的差别小,要仔细设计读出放大电路,并在读出数据后将原来的数据重新写入存储单元,或将要写入的数据写入,恢复存储电容的状态。为保证读出数据的正确,列线电平在读出前(下一次行选中)要恢复到正常读出前电平,即行选信号需要预充电时间。这样,有可能减少地址输入管脚数,实现行地址和列地址分时共用。因此 DRAM 芯片需要复杂的接口控制电路,包括行和列分时输入地址的生成以及行选和列选时序信号的生成等。DRAM 芯片的存取速度比

SRAM 芯片慢,但存储单元占用芯片的面积小,集成密度高,封装的管脚数少,每位平均价格低。DRAM 芯片用于需大容量快速存取数据的场合,例如,计算机及数字系统的主存储器、视频图像显示控制卡的显示存储器。

伪静态 RAM 芯片,又名单管 RAM 芯片,采用 DRAM 芯片的存储单元,其行和列选择的时序电路和刷新控制电路集成于芯片内。它的输入输出引脚接口与 SRAM 芯片的相近,这样便具有 SRAM 芯片接口控制简单的优点。另一方面,在速度和每位平均价格上又接近于 DRAM 芯片,最大容量在 DRAM 芯片和 SRAM 芯片最大容量之间。

RAM 芯片的容量可用 $2^N \times M$ 位表示, N 为所用地址的长度。对 SRAM 芯片, N 为地址输入引脚数;而对 DRAM 芯片, N 应为所用行地址与列地址位数之和。对大容量的 DRAM,芯片内分为 2 或 4 个存储体。 N 中还要加地址(BA)位数。 M 为字长,通常也称为芯片的输入输出数据位数。利用片选或行选、列选控制信号,可将 RAM 芯片组成阵列,在字数和字长两个方向均予扩展,以得到存储系统所需的存储容量。

随着系统对存取速度要求的提高,三种 RAM 芯片均由非同步向同步发展,同步 RAM 芯片又由单数据速率(SDR)向双数据速率(DDR)和四数据速率(QDR)发展。为了实现成组访问,行选中后,同一行地址按设定变化规则变化的不同列地址存储单元可连续访问,这样可提高系统的整体性能。

RAM 芯片的存取速度一般用读出时间表示(对同步 RAM 芯片,也可用最高工作频率表示)。对 SRAM 芯片,用地址读出时间表示;而对 DRAM 芯片,用行选读出时间(即从行地址输入有效到数据读出有效的延迟时间)表示。对非同步 RAM 芯片,这个时间可直接度量。在同样的工艺条件下,DRAM 芯片行地址读出时间总是大于 SRAM 芯片的行地址读出时间。SRAM 芯片更多地作为系统高速缓存或作为模块集成到 CPU 芯片内,而 DRAM 芯片作为系统的主存储器。成组读写适合于主存与缓存间数据的调度。

随着半导体工艺技术的发展,RAM 芯片的容量不断扩大,存取速度不断加快。满足特定需求的专用 RAM 芯片,例如先进先出(FIFO)存储器芯片、双端口 SRAM 芯片,已集成到芯片内部。目前已出现了新的专用 RAM 芯片,如以 DDR/SDR SDRAM 为基础的图像存储器芯片、网络存储器芯片、移动

SDRAM 芯片等。

参考文献

1. Rabaey J M. Digital Integrated Circuits: A Design Perspective. Prentice Hall, 1996. 北京:清华大学出版社,1998(影印版)

2. <http://www.samsung/Products/Semiconductor>

(孙祖希)

suiji cunquji

随机存取机 (random access machine, RAM) 分析算法复杂性的一种重要计算模型。现已证明,RAM 在计算能力上与图灵机等价。但在结构上,RAM 更接近通常计算机,因而,用它来分析算法的时空复杂度,结果更接近实际,也更具实用价值。

随机存取机的结构见图 1。它包括一条只读输入带,一条只写输出带,一个程序和相应的地址计数器,以及一个无限存储器。输入带分成小格,每格可放一整数。每读一个数,输入带头右移一格。输出带亦分成小格,每写一数,带头亦右移一格。RAM 的核心是程序。不同的程序代表不同的 RAM。程序是有机组成的一串指令。地址计数器标明要执行指令的地址。通常,每执行完一条指令,地址加 1。如遇转移指令,计数器内容作相应改变。存储器由无穷多个寄存器 r_0, r_1, r_2, \dots 组成。每个寄存器可放一整数。 r_0 专作累加器,各种计算在此进行。

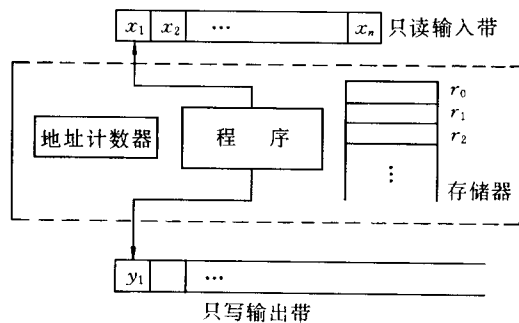


图 1 随机存取机

RAM 的指令系统包含通常计算机的各种基本算术逻辑指令。一个典型的指令系统包括 4 类 12 条指令:运算指令(加、减、乘、除),传输指令(读、写、存、取),转移指令(无条件转、零转、大于零转)和停机指令。指令可带标号,这有利于实现转移。操作数可用三种方式给定:立即数 $= i$,地址 i ,间接

地址 * i 。在指令系统中增加一般指令,既不从本质上增加计算能力,也不在数量级上改变问题的复杂度。

RAM 可作为语言识别器,但更多时候是用作实现算法的计算机。一个算法的时空复杂度是执行相应 RAM 程序时实际耗费的时间和存储空间,它们与输入数据的长度有关。这里有两种核算方式:一致方式和对数方式。在一致方式下,每条指令占一个单位时间,每个寄存器占一个单位空间。在对数方式下,时空的核算比较精确,要考虑操作类型和操作数的位数。一般采用一致方式。

RAM 是一种串行计算模型。为分析并行算法复杂度,又提出并行随机存取机 P-RAM,例如互斥读写的 EREW P-RAM,并发读互斥写的 CREW P-RAM 和并发读写的 CRCW P-RAM 等。

参考文献

Aho A V, Hopcroft J E, Ullman J D. The Design and Analysis of Computer Algorithms. Reading: Addison - Wesley, 1974 (徐美瑞)

Sunzi dingli

孙子定理 (Sun's theorem) 中国古代数学著作《孙子算经》中有“物不知其数”一问:今有物不知其数,三三数之剩二,五五数之剩三,七七数之剩二,问物几何?孙子对这个问题的解法可以表为:“三人同行七十稀,五树梅花廿一枝,七子团圆正半月,除百零五便得知。”(见程大位《算法统宗》(1593年))。它的意思为,以所求之数除以3的余数2乘70,除以5的余数3乘21,除以7的余数2乘15,总加起来得 $2 \times 70 + 3 \times 21 + 2 \times 15 = 233$,减去两个105,就得到答案23。

“物不知其数”问题就是求最小正整数 x ,使 $x \equiv 2 \pmod{3}$, $x \equiv 3 \pmod{5}$, $x \equiv 2 \pmod{7}$ 。在解法中所用到的70,21,15三个数有如下性质:70是用3除余1,用5和7都除尽的数;21是用5除余1,用3和7都除尽的数;15是用7除余1,用3和5都除尽的数。因而 $2 \times 70 + 3 \times 21 + 2 \times 15$ 是用3除余2,用5除余3,用7除余2的数。而105是3,5和7的最小公倍数。

设 $m_i (i = 1, 2, \dots, k)$ 为正整数, $a_i (i = 1, 2, \dots, k)$ 为整数,联立一次同余方程 $x \equiv a_i \pmod{m_i} (i = 1, 2, \dots, k)$,当且仅当 $a_i \equiv a_j \pmod{\text{lcm}(m_i, m_j)}$ ($i \neq j$)时才有解,且解对于模 $M (M \text{ 为 } m_1, \dots, m_k \text{ 的最小公倍数})$ 是惟一的。这就是孙子定理。上述解法也可推广到此一般情况,即先求出 $R_i (i = 1, \dots,$

$k)$,使 R_i 除以 m_i 的余数为1,而能被 $m_j (j \neq i)$ 除尽。文献上也称此定理为中国剩余定理。这个定理及其求解的思想在数学中有广泛的应用。

参考文献

华罗庚. 数论导引. 北京: 科学出版社, 1957

(裴定一)

suoyin

索引 (index) 一类特殊的数据结构,它由给定的一个或一组数据项(键码或非键码)组成,其值用来指向数据文件中相应记录的记录指针(或记录号)。其结构由两部分组成:索引数据项的值与记录指针(或记录号),见图1。

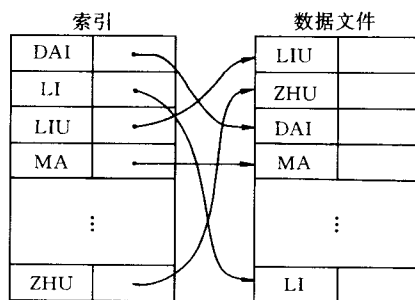


图1 索引结构示意图

基于键码建立的索引称作**主索引**,主索引的每一个索引项包含一个记录指针。基于非键码数据项建立的索引称作**辅助索引**,辅助索引的每一个数据项可包含多个记录指针。

索引的组织方式主要有线性索引和树形索引两种。在线性索引中,索引项按索引数据项的值排序,储存在顺序文件中。线性索引具有便于快速查找的优点和难于进行更新的缺点。**树形索引**是将索引组织成树形结构,树形索引既能进行快速查找,又易于索引结构的动态变化。最常用于组织索引的树形结构是**B树**及其变种(**B*树**,**B+树**等)。

在数据库的表上建立索引的作用是加快查找速度。在一个表上可以建立多个索引,以提供多种存取路径。每个索引可以建立在一个列上,也可以建立在多个列上。建立在多个列上的索引称作**组合索引**。

如果在表上建立索引时指明为**惟一性索引**,则意味着在表中不允许有两个不同的元组在索引列上取相同的值。

如果在表上建立索引时指明为**聚集索引**,则意

味着该表中的元组按索引列的值排列逻辑次序,并且元组的物理次序也与该逻辑次序相同或接近。在一个表上最多只能建立一个聚集索引。

参考文献

1. Date C J. An Introduction to Database Systems. 4th Ed. Addison - Wesley, 1986
2. Ullman J D. Principles of Database and Knowledge - Base Systems. Vol. 1. Computer Science Press, 1988
(唐世渭 杨冬青)

T

tezheng xuanze

特征选择 (feature selection) 从表征模式的有 d 个特征的初始特征集合中直接选取或通过变换, 以用较少数量的 D 个特征有效地对模式进行识别的数学方法。通常把特征的直接选取称作特征选择, 而把初始特征集合映射到较低维数特征空间的线性变换称作特征提取。它们都是统计模式识别系统中的重要组成部分, 直接关系到模式分类器的结构及性能。从初始 d 个特征中所求得的 D 个特征应具有很好的类别可分离性, 即如果用 D 维空间中的一个点表示一个模式(特征向量), 那么同类别的各个模式的特征向量应尽可能地集聚在一起, 而属于不同类的各个模式的特征向量又要尽可能地互相远离。在数学上, 我们可以定义一个表征上述要求的准则函数, 使选出来的 D 个特征相对于其他的 D 个特征组合有最大的准则函数值。对于特征选择, 有以下几种直接从初始特征集合中求取有最大准则函数值的 D 个特征组合:

(1) 枚举搜索法 从 d 个特征中组合出所有的 D 个特征集合, 分别求出相应的准则函数值, 从而选出有最大准则函数值的 D 个特征。这是一种最优算法, 其缺点是计算量大, 例如, 如果从 20 个特征中选取 10 个特征, 就要计算 $\binom{20}{10} = 184\,756$ 次准则函数值并进行大小比较, 因此一般只在特征数量很少时才采用。

(2) 次优算法 其基本出发点是只用少量的特征进行组合计算, 求出最好的特征组合后, 再不断增加新的特征, 直到得出所要求的 D 个特征。也可以从所有特征开始, 不断去掉对准则函数值贡献最小的特征, 直到减少到 D 个特征为止。理论分析表明, 次优算法不能保证得到满意的结果。

(3) 分支定界法 这是 P. M. Narendra 和 Fukunaga Keinosuke 所发展的一种最优特征选择搜索算法。由于该算法有效地组织了搜索过程, 避免了许多不影响最终结果的不必要的特征组合的运算, 在 D 比 d 小得多, 或 D 接近 d 时, 能显著地减少计算量。应用分支定界法的前提条件是准则函数的单调性, 即在已有特征基础上增加一个特征总是改善准则函数值。

对于特征提取, 如果把模式的初始特征写成向量形式, $y = [y_1, y_2, \dots, y_d]^T$, 那么通过线性变换 w , 可以得到 D 个特征所组成的新的向量 $x = [x_1, x_2, \dots, x_D]^T$, 且 $x = w^T y$, w 是 $d \times D$ 矩阵。特征提取就是在给定的表征类别分离性的准则函数下, 在所有的 $d \times D$ 矩阵中选取有最大准则函数值的 w 矩阵。

有多种度量类别分离性的准则函数。其中最常用的有基于欧氏距离度量的反映类间离散度与类内离散度关系的各种实值函数以及基于各种概率距离的准则函数等。

基于 $K-L$ 变换的特征提取方法在模式识别中得到了广泛的应用。它的基本出发点是通过 $K-L$ 变换, 消除了初始向量空间各分量之间的相关性, 从而有可能去掉那些变换后对分类没有什么作用的坐标轴以达到降低特征空间维数的目的。分析 $K-L$ 变换矩阵的各个本征值, 在给定的类别可分离性判据下, 可以从 d 个 $K-L$ 坐标轴中选取有最大判别信息的 D 个坐标轴作为模式的新的特征空间以进行分类器设计。

参考文献

1. Devijver P A, Kittler J. Pattern Recognition: A Statistical Approach. New Jersey: Prentice - Hall, 1982
2. Fukunaga K. Introduction to Statistical Pattern Recognition. 2nd ed. Boston: Academic Press, 1990

(边肇祺)

ti huizhi jishu

体绘制技术 (volume rendering technique)

由离散的三维数据场直接产生其二维图像的一种绘制技术。在这一过程中并不需要产生中间几何图元。体绘制技术的优点是能从所产生的图像中观察到三维数据场的整体和全貌, 而不只是显示出人们感兴趣的等值面(参见等值面构造技术)。同时, 也易于进行并行处理。

屏幕上的二维图像决定于缓冲存储器中对应于每一个像素点的光强度值, 它也是一个二维的离散数据场。因此, 体绘制技术的实质是将离散的三维空间数据场转换为离散的二维数据场。为此, 首先必须进行三维空间数据场的重新采样。其次, 应该

考虑三维空间中每一个数据点对二维图像的贡献,因而必须实现图像的合成。所以,体绘制技术的实现是一个三维离散数据场的重新采样和图像合成的过程。它需要很强的计算能力和较大的存储空间才能实现。

按照不同的重新采样方法,体绘制技术又可以分为按图像空间扫描和按物体空间扫描两种不同的方法。

在按图像空间扫描的体绘制方法中,根据算法对显示屏幕逐行、逐点地进行处理。当计算某一个像素点的光强度值时(参见图1),设想从该点发出一条射线,穿过数据场矩阵。沿这条射线选择若干个等距的再采样点,由距离某一采样点最近的8个数据点的函数值作3次线性插值,求出该采样点的函数值,再根据对函数值的分类结果赋予不同的颜色值及不透明度值。然后,再从前往后将每一个采样点的颜色值及不透明度值依次按下列公式迭代进行合成,得到像素点的最终颜色值。

$$\alpha_{out} = \alpha_{in} + \alpha_{now}(1 - \alpha_{in})$$

$$C_{out}\alpha_{out} = C_{in}\alpha_{in} + C_{now}\alpha_{now}(1 - \alpha_{in})$$

式中, C_{in}, α_{in} 分别代表射线进入当前再采样点之前的颜色和不透明度; C_{now}, α_{now} 分别表示当前再采样点的颜色和不透明度; C_{out}, α_{out} 分别表示叠加上当前再采样点的颜色和不透明度后的相应值。

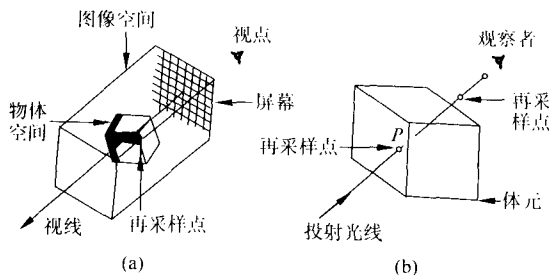


图1 按图像空间扫描的体绘制方法示意图

与按图像空间扫描的体绘制方法相反,按物体空间扫描的体绘制方法不是从图像空间的像素点出发逐点计算的,而是从物体空间的数据点出发逐点计算的。首先根据对每个数据点的函数值的分类给定不透明度值及颜色值,并给定视见平面及观察方向。然后将每个数据点的坐标由物体空间变换到图像空间,再根据选定的光照模型计算出每个数据点处的光照强度。此后,即可根据选定的重构核函数计算出由三维光照强度到二维图像的映射关系,得出每个数据点所影响的二维像素的范围及对每个像

素点光照强度的贡献。将不同的数据点对同一像素点的贡献加以合成,即可得出最后的图像。

由上所述,体绘制技术的计算量是很大的,难于在通常的PC机上实现实时绘制。近年来,采用三维纹理映射技术利用硬件可以在PC机上实现三维数据场的实时体绘制。三维纹理映射就是将三维数据场中的颜色值或光照强度视作纹理,装载入容量足够大的纹理内存,利用硬件实现三次线性插值及图像合成的并行运算,从而实现了实时体绘制。

参考文献

1. Foley J D, Dam A V, et al. Computer Graphics: Principles and Practice. Addison-Wesley Publishing Company, 1990
2. 石教英,蔡文立. 科学计算可视化算法与系统. 北京:科学出版社,1996
3. 唐泽圣. 三维数据场可视化. 北京:清华大学出版社,1999 (唐泽圣)

tiaoma yueduqi

条码阅读器 (bar code reader) 利用光电原理将条码信息转化为计算机可接受的信息的输入设备。它常用于图书馆、医院、书店以及超级市场,作为快速登记或结算的一种输入手段,对商品外包装上或印刷品上的条码信息直接阅读,并输入到联机系统中。

条码是由一组规则排列的条、空及其对应字符组成的标记,用以表示一定的信息。条是指条码中反射率较低的部分,空是指条码中反射率较高的部分。通常采用两种对比度高的颜色来分别构成条和空,例如,黑与白,蓝与黄,绿与红等等。

由于条码中条和空排列规则的不同,形成各种类型的条码,如UPC条码、EAN条码、二五条码、交错二五条码、三九条码、库德巴条码、中国标准书号(ISBN部分)条码等等。图1是按我国国家标准GB 12904—91印制的一个通用商品条码,其结构与EAN条码相同,由13位数字码以及对应的条码组成。前3位数字为前缀码,标识国家或地区(我国为690),后面4位为标识制造厂商的代码,再其后5位码是标识商品名称的代码,最后1位是校验码。条码阅读器通常带有一个发光装置,将光线照射到条码上,用光敏元件接收反射光。由于深浅不同的线条反射的光强度不同,得到高低不同的电平信号,经译码装



图1 条码

置转换为—组数字信号。译码装置和光电传感装置可以做成一体,也可以是互相独立的两个部件。

条码阅读器按工作方式可分为固定式和移动式(手持式),按光源的不同,可分为发光二极管、激光及其他光源形式。

最常见的条码阅读器是笔式阅读器,它属于手持式,俗称光笔,但与显示器屏幕光笔不是同一物品。它以发光二极管为光源,其操作如同人们用笔画一条线,只要将笔头的小窗口对准条码,在其表面匀速移动,条码信号就通过电缆进入计算机。其结构原理如图2所示。这种阅读器使用方便,价格低廉,但必须将它与条码接触。

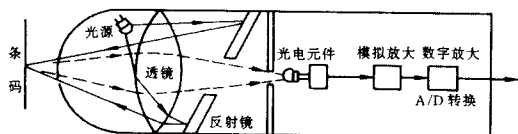


图2 笔式阅读器结构原理

卡槽式阅读器与光笔原理相同。通常是将卡槽式阅读器安装于固定的位置,而将印有条码的卡片从卡槽中划过,读取条码信息。

也可以用图像传感器来阅读条码信息。这种条码阅读器不需要与条码之间有相对运动,只要将它轻轻靠近条码便可以迅速读出条码信息。它的可靠性高,价格也稍高。

用激光作为光源的条码阅读器称为激光枪。这种阅读器阅读速度快,可实现非接触式阅读,阅读器与条码之间的距离可达500mm,但价格昂贵。

参考文献

1. 胡嘉璋. 条码国家标准应用指南. 北京: 中国标准出版社, 1991
2. 黄志建, 顾向阳, 戴均陶. 条形码技术及应用. 北京: 机械工业出版社, 1992 (程天一)

tiaoshi gongju

调试工具 (debugging tool) 软件开发过程中用于系统调试的软件工具。它辅助程序员寻找程序中错误的性质、原因并确定其位置。在此所述的调试只是针对源代码的。

已有的调试工具主要有以下两类: 源代码调试程序和调试程序生成程序。

源代码调试程序用以帮助程序员理解程序的执行状态。最初的调试工具只是对低级语言程序进行

调试, 现已逐步发展成为针对高级语言程序进行调试, 这类工具一般由执行控制程序、执行状态查询程序、跟踪包组成。执行控制程序包括断点定义、撤销、单步执行、断点执行、条件执行等功能; 执行状态查询程序包括寄存器、堆栈状态、变量、代码等与程序相关的各种状态信息的查询; 跟踪包用以跟踪程序执行过程中所经历的事件序列(如分支、子程序调用等)。程序员可通过对程序执行过程中各种状态的判别进行程序错误的识别及改正。

调试程序生成程序是一种通用的调试工具。给定一个程序设计语言, 它能生成一个相应的源代码调试程序。

参考文献

1. Leblanc J and Miller B P. Workshop Summary, Proceedings ACM SIGPLAN and SIGOPS Workshop on Parallel and Distributed Debugging. May, 1988
2. Stone, Debugging Concurrent Processes: A Case Study, Proceedings SIGPLAN '88 Conference on Programming Language Design and Implementation. July, 1988, 145 ~ 153 (钱乐秋 洪梅)

tiaozhengpan

调整盘 (customer engineer diskette) 校准磁头位置以保证软磁盘驱动器互换性的一种专用磁盘。主要用于调整磁头的径向位置和方位角。

软磁盘驱动器的互换性是指驱动器不但能正确读写本驱动器所格式化的软磁盘片, 也能正确读写其他驱动器所格式化的软磁盘片。互换性不好的主要原因是磁头定位不准确, 因此对互换性不好的驱动器需要调整其磁头的基准位置, 包括径向位置、切向位置和方位角。

检查软磁盘互换性的有各种形式的工具盘, 可以分为数字式和模拟式两大类。数字式通过微型计算机及相应的测试软件进行联机检测, 并把软磁盘驱动器的故障情况直接显示在荧光屏上。它直观, 但调整操作不方便, 适用于检查故障。模拟式用于脱机测试, 利用示波器观察波形, 易于找出故障所在, 也便于调整, 适用于维修。模拟式工具盘的全称是模拟式调整软磁盘, 简称为调整盘。由于它多用于现场维修、调整用, 因此也叫用户工程师盘, 简称为CE盘。有的使用手册根据读出信号波形, 称之为猫眼盘。CE盘是常用的调整磁头基准位置的工具盘。

调整盘是用专用设备制造的, 不能用一般软磁盘驱动器复制。以5.25英寸高密度的调整盘为例,

它的盘面上录有以下几种信号：①在零道上录有 250 kHz 的准正弦波连续信号；②在 02 道有索引脉冲串；③在 32 道上是猫眼信号；④在 64 道上有 125 kHz 的准正弦波；⑤在 68 道上有和 02 道一样的索引脉冲串和方位角脉冲串；⑥在 66 道和 79 道上有和零道一样的 250 kHz 的准正弦波。

在 32 道上的猫眼信号用来调整磁头的径向位置,由专用录制设备录制。录制时软磁盘片的物理中心 O 和主轴的中心 O' (即盘片的旋转中心)有一偏心值,距离为 e 。在录制设备上的磁头在正确位置两侧各录一个磁道,两磁道中间间隔为 Δ 。 e 和 Δ 的数值接近于磁道宽度。这两个磁道对中心 O' 来说是同心圆。当录好的软磁盘片装入普通的驱动器时,这两个磁道对中心 O 来说是偏心的。因此在正确磁道位置的磁头不可能读出完整的一圈信号,而只能读出两磁道的各自一半,如图 1 所示。在示波器上看到的波形如图 2 所示,图中左右波形分别为

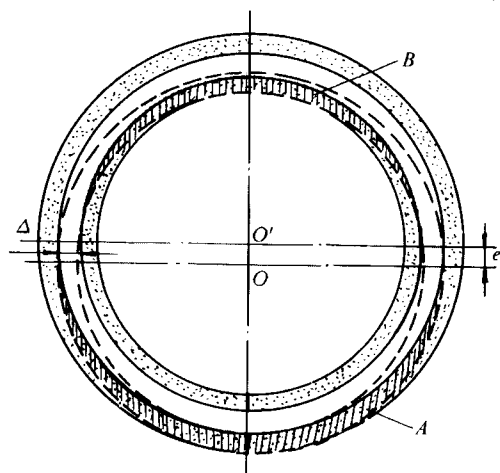


图1 猫眼信号形成原理

U_A, U_B 。当磁头位置正确时,外侧磁道A、内侧磁道B两磁道各有半圈被读出,此时信号幅度 $U_A = U_B$,如图2(a);当磁头位置向外偏离正确位置时,读出信号幅度 $U_A < U_B$,如图2(b);反之,向内偏离正确位置时,读出信号幅度 $U_A > U_B$,如图2(c)。根据信号大小情况调整磁头径向位置。一般 $U_A / U_B > 70\%$,或 $U_B / U_A > 70\%$ 时就需要调整。由于在示波器上看到的信号是大小变化的两个近似正弦波的信号,类似猫眼,称之为“猫眼信号”。

在 68 道上的方位角脉冲串用来调整磁头的方位角。磁头的方位角指磁头的前间隙与磁盘半径间

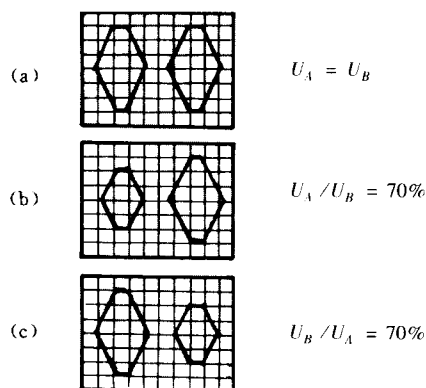


图2 猫眼信号波形

的夹角。标准状态是方位角等于 0,这时从 68 道上读出的波形如图 3(a) 所示,4 个脉冲串中 2,3 的幅度相等,1,4 的幅度相等。图 3(b) 是方位角有正偏差的情况,图 3(c) 是方位角有负偏差的情况。出现这两种偏差都需要调整磁头的方位角。

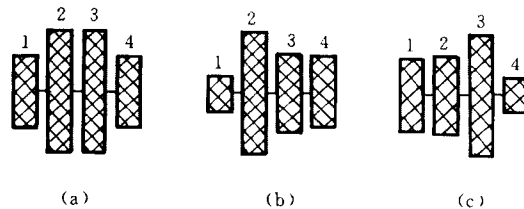


图3 方位角脉冲串

在零道上的信号供调整磁头零道位置用。64 道上写的是基准频率的信号,66 道上写的是 2 倍基准频率的信号,供测试分辨率用。在 79 道上的信号用于检查步进精度。在 02 道上的索引脉冲串用于检查调整索引信号与磁头间的距离(角度)。在 68 道上的索引脉冲串用来检查磁头运动的轨迹与磁盘半径的夹角。当磁头运动轨迹是磁盘半径方向时,夹角等于 0,用 02 道和 68 道的索引脉冲串测得的时间相等。

调整盘作为一种工具,不同的磁盘标准如 5.25 英寸双面倍密度、3.5 英寸 1 MB 及 3.5 英寸 2 MB 等都有相应的产品供选择使用。不同厂家生产的调整盘信号略有不同,使用时应注意。

调整盘使用前应在使用的环境下放置 24 h 以上,以适应环境。调整盘不得写入信号,也要避免误操作造成写入,例如把调整盘插在软磁盘驱动器中开关电源时,驱动器写保护或写电路有故障时都有可能发生误写入。另外,调整盘价格较贵,应避免划

伤、夹坏等事故发生。

参考文献

朱传乃,刘镜周. 80286 微型计算机的原理与维修. 北京: 科学出版社, 1992 (王国元)

tiaozhi jietiaoqi

调制解调器 (modem) 可将数字信号转换成模拟信号,以适于在模拟信道中传输,又可将被转换的模拟信号还原为数字信号的设备。它将计算机或各种终端设备与模拟信道(如通信线路)相连接,以便计算机之间,计算机与工作站之间或其他外围设备之间传输信息。modem 一词是由 modulation(调制)和 demodulation(解调)两个词复合而成的。

利用公用电话网传输计算机用的数字信号,优点是方便和经济。但电话线路是为传输语音模拟信号而设计的,语音的频带宽度为 300 Hz ~ 3 400 Hz,用来传输数字信号会引起信号的严重衰减和失真。解决的办法是把数字信号在发送前变换成模拟信号(称为“调制”),在接收端再把模拟信号变换成数字信号(称为“解调”)。调制解调器就是进行这种变

换的设备。由于通信是双方向的,因此通常把调制器和解调器做在一起,一个设备既有调制的功能,也有解调的功能,所以称为调制解调器。

调制在通信上应用已久,它用一定频率的正弦波作载波,用另一个波控制载波参数的变化。正弦波信号有幅度、相位、频率等三个参数,因此调制也有幅度调制、相位调制、频率调制等三种基本方式。解调则相反,从已调制的正弦波中分解出原来的控制波。调频广播、调幅广播是调制应用的典型例子。在计算机通信中,仍是用一定频率的正弦波作载波,但控制的是数字信号,即电平信号。图 1 画出了二进制调幅、调频、调相的波形图。通常把幅度调制称为**幅移键控 (ASK)**, 相位调制称为**相移键控 (PSK)**, 频率调制称为**频移键控 (FSK)**。其中幅移键控最简单,但性能也差,易受干扰。频移键控对干扰不敏感,但要求信道的频带宽一些。相移键控有两种基本形式:以载波相位为参考点的称为二进制相移键控(BPSK),以前一个码元的相位为参考点的称差分相移键控(DPSK)。相移键控的信道频带要比频移键控的宽,实现的电路也复杂,但其抗干扰能力比幅移键控、频移键控都好得多。

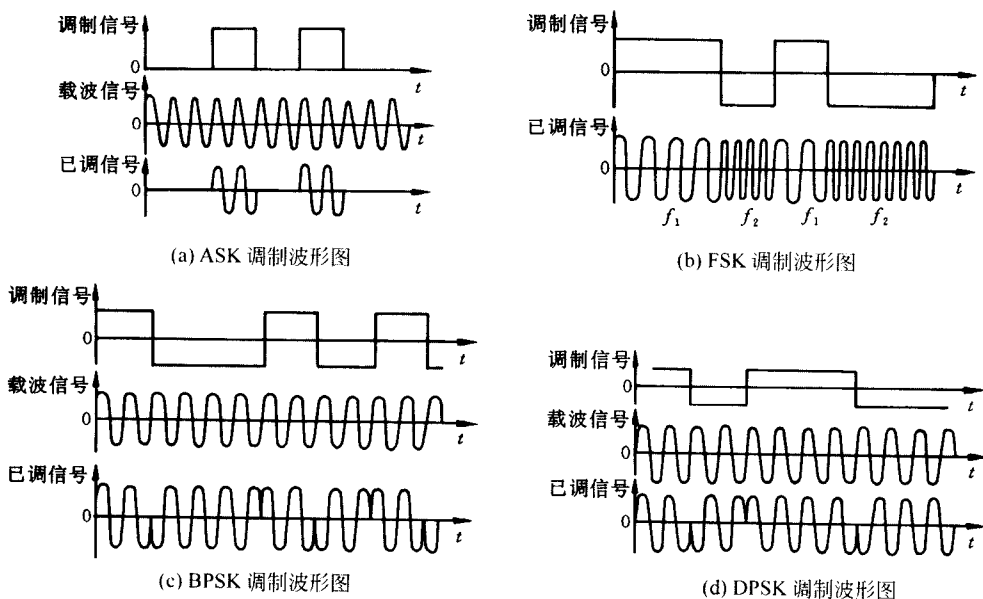


图 1 二进制调制波形图

调制只是改变信息的载体,并没有改变信息的内容。通常把在通信线路中携带信息的信号变化称为“码元”,每秒通过线路的码元数称为码元传输速

率,单位是波特。每秒钟通过线路的二进制信息量称为位传输速率,也称比特率,单位为 b/s。在性能一定的通信线路中,要提高信息传输速率必须使载波信

号参数每一次的变化(每一个码元)能携带更多的信息量。多进制调制就有这样的能力。例如,对二进制FSK来说,用 f_0 表示0,用 f_1 表示1,码元变化一次只代表一个二进制信息。如果采用四进制FSK,其编码表如表1所示。从表可见,每一个码元代表了两位二进制信息。在同样的波特情况下,比特率提高了一倍。更常用的是对幅度和相位同时调制,即正交调幅(QAM)。虽然正交调幅在设备上比较复杂,但它在提高比特率的同时降低了误码率。正交调制是目前中高速调制解调器常用的一种调制方式。20世纪80年代初,提出了网格编码调制(TCM),把调制技术和纠错编码结合在一起,既保证了高信息传输速率,

又能降低误码率,比特率可达14 400 b/s。

表1 四进制调制编码表

载波频率	f_1	f_2	f_3	f_4
信息编码	00	01	10	11

调制解调器是在通信线路上使用的,必须有统一的标准。国际电报电话咨询委员会(CCITT)对调制解调器制定了一系列的建议标准,对使用与制造提出了统一的要求,如规定选用波特的值应为600的整数倍等。与计算机通信有关的一些CCITT建议列于表2。

表2 CCITT关于调制解调器的若干建议

CCITT 建议号	比特率 / b/s	说 明
V. 21	300	全双工, 公用交换电话网
V. 22	1 200	全双工, 公用交换电话网, 二线租用电话电路
V. 22bis	2 400	全双工, 公用交换电话网, 二线租用电话电路
V. 23	600 / 1 200	半双工, 公用交换电话网
V. 26	2 400	全双工, 四线租用电话电路
V. 26bis	2 400 / 1 200	全双工, 公用交换电话网
V. 26ter	2 400	全双工, 公用交换电话网, 二线租用电话电路
V. 27	4 800	全双工, 四线租用电话电路
V. 27bis	2 400 / 4 800	全双工, 四线租用电话电路
V. 27ter	4 800 / 2 400	全双工, 公用交换电话网
V. 29	9 600	点对点四线租用电话电路
V. 32	9 600	全双工, 公用交换电话网, 点对点二线租用电话电路
V. 33	14 400	点对点四线租用电话电路

注: bis —— 第二版或修订版; ter —— 第三版

调制解调器可按照不同的角度分类。按传输速率可分为低速(1 200 b/s 以下), 中速(2 400 ~ 4 800 b/s), 高速(9 600 b/s 以上)。按调制方式分为FSK, PSK, QAM, TCM等。此外, 还可分成并行和串行, 同步和异步, 公共电话网和专用线路, 机内和外挂, 单工、双工和半双工, 长距离和短距离, 等等。

第一台便携式调制解调器是1967年上市的, 1980年研制出采用QAM调制技术的智能型调制解调器。随着计算机技术与大规模集成电路技术的发展, 低速的调制解调器已做成芯片, 把调制、解调、控制、滤波、锁相等电路都集成在一块芯片上。各种新型的调制解调器, 如无线调制解调器、多载频调制解调器等都已出现。现在许多调制解调器内有微处理器, 因而具有自动拨号、自动应答、自动纠错、自动降速、自诊断等功能, 高速调制解调器还加有纠错功能和数据压缩功能。

今后, 计算机技术与通信技术的结合越来越紧

密, 调制解调器的应用也越来越广泛, 将会出现体积更小、功能更多、更安全可靠的新品种。

参考文献

于康友等. 调制解调器原理、选用和测试. 北京: 电子工业出版社, 1994

(林兼)

tingji wenti

停机问题 (halting problem) 一个重要的不可判定问题, 由它可以推出计算机科学中的许多不可判定问题。一般性的停机问题指是否存在这样的算法, 使得对于任意的图灵机 M 和任意输入 x , 可以判定对输入 x , 图灵机 M 能否停机。早在1936年A. Turing证明了停机问题的不可判定性, 他在证明一般性的停机问题的不可判定性时, 采用了康托对角线化方法。对角线化方法已成为递归论和计算复杂性理论等学科中重要的工具, 其中许多深刻的结论是通过这一方法获得的。由一般性的停机问题的

不可判定性和通用图灵机的存在性,立即可推证特定图灵机的停机问题也是不可判定的。特定图灵机的停机问题指是否存在这样的算法,使得对于给定的图灵机 M 和任意的输入 x ,可以判定对输入 x ,图灵机 M 能否停机。停机问题中的另两个重要的不可判定问题是图灵机的完全性和等价性判定问题。图灵机的完全性问题指是否存在这样的算法,使得对于任意图灵机 M ,都可以判定 M 是否对任何输入都停机;图灵机的等价性问题指是否存在这样的算法,使得对于任何两个图灵机,都可以判定是否对任何输入这两个图灵机都停机,且有相同输出,或者都不停机。这类停机问题在程序设计学中均可找到实际的解释。

参考文献

1. 张鸣华. 可计算性理论. 北京: 清华大学出版社, 1984
2. 李祥. 可计算性理论导引. 贵州人民出版社, 1986
(马绍汉 李大兴)

tongxin fuwuqi

通信服务器 (communication server) 为需要通过远程通信链路传送文件、访问远地系统及网上信息的用户提供通信服务的专用服务器。

通信服务器可提供下列一个或多个服务功能:

- (1) 网关功能 通过转换数据格式、通信协议和电缆信号提供用户与主机的连接。
- (2) 访问服务 允许远地用户经拨号进入网络。
- (3) 调制解调器 通信服务器为用户提供一组异步调制解调器,用于拨号访问远地系统、信息服务系统或其他资源。
- (4) 网桥或路由器功能 维持与远地局域网的专用或拨号链路,并在局域网间自动传送数据分组。
- (5) 电子邮件服务 自动连接其他局域网或电子邮箱,收集和传递电子邮件。

参考文献

- Sheldon T. LAN Times Encyclopedia of Networking. McGraw-Hill Inc., 1994
(胡道元)

tongxin kongzhiqui

通信控制器 (communication controller)

通过硬连线把若干个远程终端与主计算机相连接的一种通信控制设备。通信控制器与前端处理器的功能是相同的,不同的是前端处理器用计算机编制程序来实现,而通信控制器则是主计算机中的一个硬

连线部件。

随着所实现的功能及执行的通信任务的不同,通信控制器的构成也不尽相同,典型的通信控制器的构成如图 1 所示。图中线路连接单元(LCU)的作用是把数据发送到通信线路上,或接收来自通信线路的数据。它由进行电平变换的线路驱动器 and 收发共用器等组成。

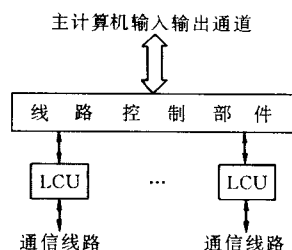


图 1 一个典型的通信控制器的构成

线路控制部件在与多个线路连接单元进行周期性的数据字符交换的同时,还要识别传送控制符,生成和检测纠错码以及经由输入输出通道与主计算机进行数据传送等。这些工作都是采用微程序控制由硬件来完成的。
(过介望)

tongxin kongzhi shebei

通信控制设备 (communication control unit)

计算机网络中控制发送和接收数据的设备。通常,这种设备是用来对网络中的计算机、工作站、终端或其他数字设备进行互连的。它可以是一个复杂的设备,例如前端处理器,也可以是一个较为简单的设备,例如网桥。

由于在计算机内部的数据字符以位并行方式传送,而在通信线路上则是以位串行方式传送,因此通信控制设备必须完成下列基本功能: ①在发送端把数据从并行形式转换到串行形式,亦即并串转换; ②在接收端把数据从串行形式转换到并行形式,亦即串并转换; ③为了实现数据同步、差错控制等,需要加上或除去控制比特或控制字符。

随着计算机网络的规模和复杂性的增加,越来越多的以前由主计算机完成的功能,转由通信控制设备来完成。为此,通信控制设备还需具备以下功能: ①线路探测; ②自动速度(波特)检测; ③自动拆接不工作的设备; ④支持许多各不相同的协议,并进行它们之间的转换; ⑤键盘映射和代码转换(例如从 ASCII 码转换到 EBCDIC 码,或相反); ⑥报文的缓冲、存储和转发; ⑦传输差错检测和校

正;⑧数据压缩;⑨报文的路由选择和筛选;⑩安全控制;⑪网络业务量统计数据的收集;⑫最佳路由的选取;⑬传输差错和网络故障的记录;⑭网络诊断程序的运行;⑮万一线路或设备发生严重故障,进行低效运行的切换。

按照互连的情况及所完成的主要功能,通信控制设备目前已有前端处理器、通信控制器、网关、路由器、网桥、中继器、网络适配器和终端服务器等多种。图1示出了通信控制设备在计算机网络中所提供的某些典型的功能支持。

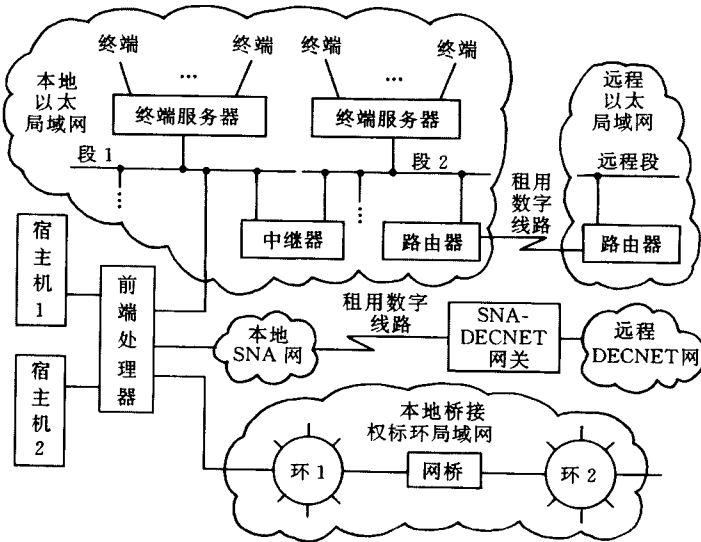


图1 通信控制设备提供的典型功能支持

通信控制设备在早期几乎都是硬连线的控制器,现在绝大多数已采用程序控制的微型计算机,通常包含一个中央处理器(CPU),它控制着许多线路单元(LU)和一些任选的输入输出(I/O)处理器,如图2所示。

CPU和它的存储器是通信控制设备的核心,通

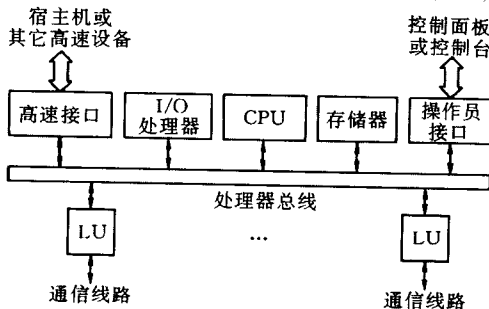


图2 通信控制设备的一个典型构成

过程序来完成通信功能并控制各种线路单元、I/O处理器和相关接口的全部操作。CPU完成较复杂的功能,诸如寻址、对话建立、差错记录,在某些严重故障的情况下进行低效运行的切换。有些更为常规的或基本的功能则由CPU分配给别的部件来执行。

线路单元负责把数据发送到通信线路上或接收来自通信线路上的数据,在CPU的控制下完成诸如数据的并串转换或串并转换、分组的装配或拆卸、同步以及差错检测和校正等功能。

大部分通信控制设备具有一个操作员接口,用

来监视通信控制设备的工作,以及必要时启动诊断程序来进行测试。有些通信控制设备还设有一个或多个I/O处理器,以与主计算机(在这种情况下又称为宿主机)或大容量磁盘等外围设备进行高速的数据传送。

参考文献

Tanenbaum A. S. Computer Networks. 2nd ed. Prentice - Hall, 1988
(过介整)

tongxin shunxu jincheng
通信顺序进程 (communicating sequential processes, CSP) 用于描述分布式通信系统的语言,首倡者是英国学者 C.

A. R. Hoare。有两个不同而又相互联系的语言通常都称为 CSP。下面分别加以介绍。

1978年 C. A. R. Hoare 提出的通信顺序进程 CSP,是面向分布式系统的程序设计语言。在该语言中,一个并发系统由若干并行运行的顺序进程组成,每个进程不能对其他进程的变量赋值。进程之间只能通过一对通信原语实现协作: $Q?x$ 表示从进程 Q 输入一个值到变量 x 中; $P!e$ 表示把表达式 e 的值发送给进程 P 。当 P 进程执行 $Q?x$,同时 Q 进程执行 $P!e$ 时,发生通信, e 的值从 Q 进程传送给 P 进程的变量 x 。后来出现的实用编程语言 OCCAM 即以 CSP 为基础发展而成。

1984年 S. Brooks, C. A. R. Hoare 和 W. Roscoe 提出 CSP 理论 (TCSP)。这是一个代数演算系统,其基本成分是事件(或动作)。进程由事件和一组算子

构造而成。典型的算子有： \rightarrow （前缀）， $|$ （外部非确定性选择）， \sqcap （内部非确定性选择）， \sqcup （交错并行）， \parallel （同步并行）， $\backslash e$ （事件隐蔽），以及递归等。

例：（自动售货机）

$VM = \text{coin} \rightarrow (\text{choc} \rightarrow VM \mid \text{coffee} \rightarrow VM),$

$CUST = \text{coin} \rightarrow (\text{choc} \rightarrow CUST \sqcap \text{coffee} \rightarrow CUST)$

这里定义了两个进程：VM（售货机）和 CUST（顾客）。售货机在接受了硬币 coin 后，可按顾客的要求支付 choc 或 coffee。顾客在付了硬币后，或者想要 choc，或者想要 coffee，其选择不受外界影响。

与 CCS 不同（也与作为程序设计语言的 CSP 不同），TCSP 采用的是广播式通信，而不是握手式通信，即只有当并行运行的各进程都执行同一动作时，才发生同步。

TCSP 采用失败等价作为确定进程等价的准则，这也称为失败语义。一个失败是一二元组 (s, X) ，其中 s 是事件的有限序列， X 是事件集（称为拒绝集）。若进程 P 可以执行事件序列 s ，且到达一个无法执行 X 中任一事件的状态，则称 (s, X) 是 P 的一个失败。例如，设有进程 $P = a \rightarrow (b \rightarrow c \rightarrow \text{STOP} \mid b \rightarrow d \rightarrow \text{STOP})$ ，则 $(ab, \{c\})$ 是 P 的一个失败，因为 P 可执行 ab ，变成 $d \rightarrow \text{STOP}$ 而无法做 c 。类似地， $(ab, \{d\})$ 也是 P 的一个失败，但 $(ab, \{c, d\})$ 不是。两个进程失败等价当且仅当它们具有相同的失败集。例如对自动售货机的例子，可以证明进程 $VM \mid CUST$ 与 $CUST$ 失败等价。

利用失败可以构造 TCSP 的指称模型，在此模型中，失败等价的进程被解释为同一个元素。关于失败等价建立了一些公理系统，可以对语义上的等价关系进行形式推导。

参考文献

1. 陆汝铃. 计算机语言的形式语义. 科学出版社, 1992
2. Hoare C A R. Communicating Sequential Processes. Comm. ACM, 1978, 21(8): 666 ~ 677
3. Brooks S D, Hoare C A R and Roscoe A W. A Theory of Communicating Sequential Processes. J. ACM, 1984, 31(3): 560 ~ 599 (林惠民)

tongxin xitong yansuan

通信系统演算 (calculus of communication systems, CCS) 英国学者 R. Milner 提出的用于描述通信并发系统的代数理论。

假定一个标号集 L ，其补集 $\bar{L} \stackrel{\text{def}}{=} \{\bar{a} \mid a \in L\}$ 。

$Act = L \cup L \cup \{\tau\}$ 称为动作集，其中 τ 是特殊的不可见动作。CCS 的进程构造算子如下：

算子	直观意义
0	空进程
$a. P$	动作前缀 ($a \in Act$)
$P + Q$	非确定选择
$P \mid Q$	并行复合
$P \backslash S$	限制 ($S \subseteq L$)
$P[f]$	换标号 (f 是从 L 到 L 的部分函数)

此外还允许递归算子。

CCS 的语义由结构化操作语义方法给出，下面列出几条典型的语义规则。 $P \xrightarrow{a} Q$ 表示进程 P 可执行动作 a 而演变为 Q 。

$$\begin{array}{lll}
 \text{ACT } a. P \xrightarrow{a} P & \text{SUM1 } \frac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'} & \text{SUM2 } \frac{Q \xrightarrow{a} Q'}{P + Q \xrightarrow{a} Q'} \\
 \text{PAR1 } \frac{P \xrightarrow{a} P'}{P \mid Q \xrightarrow{a} P' \mid Q} & \text{PAR2 } \frac{Q \xrightarrow{a} Q'}{P \mid Q \xrightarrow{a} P \mid Q'} & \text{COM } \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{a} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'} \\
 \text{RES } \frac{P \xrightarrow{a} P'}{P \backslash S \xrightarrow{a} P \backslash S} \quad a \notin S \cup \bar{S} & \text{REL } \frac{P \xrightarrow{a} P'}{P[f] \xrightarrow{f(a)} P'[f]} &
 \end{array}$$

从关于并行算子 $|$ 的规则可以看出，由两个进程并行复合而成的进程可以做每个分进程所能做的动作 (PAR1) 和 (PAR2)，但当两个分进程同时执行一对互补的动作时，则发生通信，产生 τ 动作 (COM)。这种通信方式称为握手式通信。CCS 的基本思想是用 τ 和 $+$ 来模拟 $|$ ，将并发归结为非确定性，即所谓交错语义。这一思想体现在下面的展开律中：

$$Q_1 \mid \cdots \mid Q_n = \sum \{ \alpha. (Q_1 \mid \cdots \mid Q'_i \mid \cdots \mid Q_n) : i \leq n,$$

$$Q_i \xrightarrow{\alpha} Q'_i \} + \sum \{ \tau. (Q_1 \mid \cdots \mid Q'_i \mid \cdots \mid Q'_j \mid \cdots$$

$$\mid Q_n) : i < j \leq n, Q_i \xrightarrow{i} Q'_i, Q_j \xrightarrow{j} Q'_j \}$$

CCS 采用互模拟作为基本的进程等价关系。强互模拟等价满足下面的 Monoid 公理：

$$P + 0 = P, P + P = P, P + Q = Q + P,$$

$$(P + Q) + R = P + (Q + R)$$

对观察等价（将 τ 忽略不计）还成立三条 τ -公理：

$$a. \tau. P = a. P,$$

$$P + \tau. P = \tau. P,$$

$$a. (P + \tau. Q) + a. Q = a. (P + \tau. Q)$$

上面介绍的 CCS 称为基本 CCS，或纯 CCS，进程之间只能通过执行互补动作实现同步，而不能直接进行通信。全 CCS 引入输出动作 $c!e$ 和输入动作 $c?x$ ，

$c!e$ 表示沿通道 c 发送数据表达式 e 的值, $c?x$ 表示从通道 c 接收一个值赋给 x , 此外还有条件表达式 if-then-else 。用全 CCS 可以直接描述进程间的通信。

参考文献

1. 陆汝钤. 计算机语言的形式语义. 北京: 科学出版社, 1992
2. Milner R. Communication and Concurrency. New York: Prentice - Hall, 1989 (林惠民)

tongyong duo bawei bianma zifuji ISO/IEC 10646

通用多八位编码字符集 ISO/IEC 10646 (Universal Multiple-Octet Coded Character Set—ISO/IEC 10646, UCS)

国际标准化组织 ISO/IEC 制定的旨在实现全球所有文字符号统一编码的一项重要国际标准。ISO/IEC 已经发布了该标准第 1 部分的第 2 版 (ISO/IEC 10646—1:2000) 和第 2 部分的第 1 版 (ISO/IEC 10646—2:2001)。我国与 ISO/IEC 10646 等同的国家标准是 GB 13000。

字符集及其编码是计算机中表示、储存、处理和交换文本信息的基础。有关统计表明, 目前世界各个国家和民族几乎有 6 800 种不同的语言和文字在使用。随着经济全球化趋势的加快, 使用计算机处理、存储和传输任意多种语言文字的需求日益迫切, 因而必须为计算机建立一个多文种信息处理环境。

许多年来, 绝大多数计算机系统所采用的字符集都是以国际标准 ISO/IEC 2022 为基础的。ISO/IEC 2022 定义了七位代码和八位代码的空间及代码空间的扩充技术, 即除了标准 ASCII 字符集之外, 还有其他几百种不同的扩充字符集 (包括 GB 2312 和 GBK 汉字字符集)。不同字符集各有一个惟一的代码页号。当文本中出现非 ASCII 字符时, 先使用代码页号指出它属于哪一个字符集, 然后才是该字符在字符集中的编码。由于这些字符集没有哪一个可以适用于所有的字母、标点符号和常用的技术符号, 其编码还会互相冲突 (不同字符集中可能使用相同的代码代表两个不同的字符, 或使用不同的代码代表相同的字符), 因此不仅使用比较繁琐, 而且在不同的系统中交换数据时, 总会有损坏的危险。

解决上述问题的方法是对所有字符进行统一编码, 即不论是什么计算平台, 不论是什么程序, 不

论是什么语言, 每个字符都使用一个惟一的编码。ISO/IEC 10646 标准 (UCS) 规定, 全世界现代书面文字所使用的所有字符、符号都使用四字节进行编码 (记作 UCS-4)。这样做的优点是编码空间极大 (图 1), 可以安排 13 亿个字符, 能容纳足够多的各种字符集, 充分满足世界上多种民族语言文字信息处理的要求。

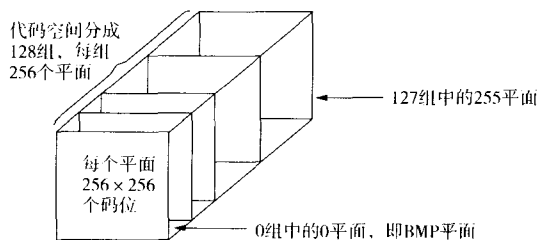


图 1 UCS 的代码空间

可是, 四字节的字符编码太浪费存储空间了。比较实际的做法是, 在 UCS 编码空间中, 把第 1 和第 2 字节均为“0”的一个子空间 (称为基本多文种平面 BMP), 作为它的子集来使用, 记作 UCS-2 (-2 表示双字节编码)。在 UCS 国际标准的第 1 部分 (ISO/IEC 10646—1:2000) 中, 规定了 UCS-2 的全部字汇和编码, 其中包含世界各国和地区当前主要使用的拉丁字母文字、音节文字和汉字中的常用字以及各种符号和数字共 49 194 个, 主要有: ①欧洲及中东地区使用的拉丁字母、音节文字; ②各种标点符号、数学符号、技术符号、几何形状、箭头及其他符号; ③中日韩统一汉字 (CJK 汉字) 20 902 个和 CJK_A 扩充的 6 582 个中日韩汉字。

UCS 国际标准的第 2 部分 (ISO/IEC 10646—2:2001) 则进一步定义了 01 平面 (文字与符号辅助平面, 简称 SMP 平面)、02 平面 (CJK 汉字辅助平面, 简称 SIP 平面) 和 0E 平面 (特殊用途平面, 简称 SPP 平面) 的内容。其中, 02 平面码位在 0002 0000 ~ 0002 A719 范围内的是 CJK-B 扩充的共 42 778 个中日韩汉字。

为了与目前大量使用的基于 ISO/IEC 2022 的单八位系统保持向下兼容, 同时避免与数据通信中使用的控制码发生冲突, UCS 在实现时可以将双字节代码变换为可变长代码, 最常用的一种称为 “UTF-8” 形式 (UCS Transformation Form-8), 它按照表 1 给出的规则, 把双字节的 UCS-2 编码转换为单字节、双字节或三字节和四字节的 UTF-8 编码。

表 1 UCS-2 编码到 UTF-8 编码的部分转换规则

UCS-2 编码		UTF-8 编码	
范围	转换前的代码	转换后的代码	字节数
0000 ~ 007F hex	00000000 0xxxxxxx	0xxxxxxx	1
0080 ~ 07FF hex	00000xxx xxyyyyyy	110xxxxx 10yyyyyy	2
0800 ~ FFFF hex	xxxxyyyy yyzzzzzz	1110xxxx 10yyyyyy 10zzzzzz	3

从表中可以看出,标准 ASCII 字符仍以单字节代码(00H~7FH)表示,其他字符如 CJK 汉字和扩充的拉丁字母、音节文字、标点符号等,需要使用双字节、三字节或四字节代码表示。这样,既保持了与传统 ASCII 文本兼容,避免了与数据通信中控制码的冲突,又实现了各种字符集的统一编码。目前,大多数 UCS 编码都是以 UTF-8 编码形式实现的。

UCS 给世界各国和地区使用的主要字符提供了统一编码的解决方案,国际标准 ISO/IEC 10646 的制定和相应工业标准 Unicode 的成功实现,是近来全球软件技术最重要的发展趋势。UCS/Unicode 标准已经被 IBM, Microsoft, Apple, HP, Oracle, SAP, Sun 等公司采用,许多最新的软件标准都需要 UCS/Unicode,例如 XML, Java, ECMAScript (JavaScript), LDAP, CORBA 3.0, WML 等。

参考文献

1. <http://www.unihan.com.cn/cjk/cjkhome.htm>
2. ISO/IEC 10646—1: Universal Multi-Octet Character Set—UCS-Part 1: Architecture and Basic Multilingual Plane (张福尧)

tongyong jicunqi

通用寄存器 (general purpose register) 中央处理器内部用于在处理过程中临时存放中间结果和参数的一种多用途寄存器。它由与中央处理器其他逻辑电路同类的电路组成,运行速度快,可以被看作是比**高速缓冲存储器**速度更快的一个存储层次,但它不与存储器统一编址。

由于超大规模集成电路的发展,通用寄存器已由过去用若干个集成电路芯片所组成,发展为整体地集成在中央处理器芯片中。

通用寄存器用于存放操作数以及用作**累加器**、**变址寄存器**、**基地址寄存器**、**自动增减量寄存器**、**栈指示器**等。不同的计算机对通用寄存器的使用方式和配置数量可有很大差别。这些寄存器可以为各种用途共同使用,也可以有所侧重。但无论如何,涉及通用寄存器的指令必须包含寄存器的号码和使用方

式码。前者用以选定某一个寄存器,后者指定该寄存器的用途。有的指令不含使用方式码也必然有别的途径指明其用途。

寄存器与寄存器之间进行操作的指令运行速度最快,从这个意义上讲通用寄存器的数量越多越有利。但另一方面,过程调用时必须对现场进行处理,通用寄存器与存储器之间需要进行频繁的读写,从这个意义上讲通用寄存器的数量又不宜太多。另外,集成电路工艺水平也是制约通用寄存器配置数量的重要条件。目前多数计算机配置通用寄存器的数量在 8 至 16 个左右,这是考虑各种制约因素后的折中。

精简指令集计算机的结构比一般计算机简单,中央处理器芯片可以腾出足够的面积容纳较多的通用寄存器。有的精简指令集计算机利用这个条件配置了数以百计的通用寄存器,但每一个程序过程只能接触其中若干个寄存器,这些寄存器称为窗口。全部通用寄存器分为若干个窗口,一个过程使用一个窗口。相邻窗口有几个寄存器是相互覆盖的,这样就为沟通调用和被调用的两个过程之间的信息提供了方便。另外还有若干个寄存器存放全局性参数供各个过程共用。这样的使用方式大大减少了访问存储器的次数,使大多数操作都在寄存器与寄存器之间进行,从而提高了中央处理器的运行效率。

通用寄存器是从属于中央处理器体系结构的一个部件。随着电路集成度的不断提高和体系结构技术的改进,通用寄存器配置数量的制约将放宽,使用方式的设计自由度将扩大。(张梓昌)

tongbu chuanshu

同步传输 (synchronous transmission) 采用共同时钟定时机制达到发送方和接收方的时钟信号匹配的数据传送方式。同步传输的关键是发送方和接收方必须使它们的时钟同步,这样接收方才能精确地知道新的字符从哪儿开始。为了维持长时间的时钟同步,一个特定的位转换模式被嵌入数字信号,以帮助维持发送方和接收方之间的定时。

在同步传输的情况下,为了使接收方确定数据

块的开始和结束,每个数据块用一个前文位模式开始,用一个后文位模式结束。加有前文和后文的数据称为一帧,前文和后文的特性取决于数据块是面向字符的还是面向位的。

如果采用面向字符的方案,每个数据块以一个或多个同步字符作为开始,如图 1 所示。同步字符通常称为 SYNC,它的选择依据是其位模式与传输的任何正规字符都有明显的差别。后文是另一个惟一的字符,这样,接收方就要注意由 SYNC 字符引导而进入的数据块,并且接收数据,直到发现后文字符为止,然后接收方就寻找下一个 SYNC 字符。

同步 字符 (SYNC)	8 位 地址域	8 位 控制域	数据域	循环冗余 校验位	数据 结束 标志
--------------------	------------	------------	-----	-------------	----------------

图 1 面向字符的同步传输

现在,诸如 IBM 的二进制同步规程 BISYNC 一类的面向字符的方案正在逐步地被更有效、更灵活的面向位的方案所替代。这种方案把数据块作为位流来处理,而不是作为字符流来处理。除了前文和后文的原理与面向字符的方案有一点差别外,其他完全相同。由于该方案假设数据是一个任意的位模式,因此不能够保证在数据中不发现前文或后文。

两个最普通的面向位的方案是 HDLC 和 SDLC,它们把模式 01111110(称为标志)既作为前文使用也作为后文使用。为了避免在数据流中出现这种模式,发送方总是在所发送的数据中每出现 5 个 1 之后就插入一个附加的 0,当接收方检测到 5 个 1 的序列时,就检查以后的一位数据,若该位是 0,接收方就删除它,这种规程就是所谓的位插入。

参考文献

1. 胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002
2. Sheldon T. LAN Times Encyclopedia of Networking. McGraw-Hill Inc., 1994 (胡道元)

tongbu guangqian wang

同步光纤网(synchronous optical network, SONET) 一种基于光纤作为传输介质的同步传输网络。美国国家标准委员会(ANSI,即 NISI)为其制定了光传输接口标准。与其相当的是国际电信联盟 ITU-T(即原 CCITT)制定的同步数字系列(SDH)光传输标准。这两个标准本质上是等同的

和相互兼容的。同步光纤网(SONET)定义了一个信号的层次结构及格式,称为同步传输信号(STSs),及其相应的物理链路的光纤载体(OCs)。STSs 定义了从 STS-1 到 STS-192,每个层次支持一个特定的数据速率;OCs 定义了相应的链路概念和物理规范,从 OC-1 到 OC-192。SDH 则定义了一个称为同步传输模块(STM)的类似系统,与 STSs 系统兼容,也与欧洲现行的体系如 E 链路兼容。表 1 给出了 SONET/SDH 的相应的速率对照表。

表 1 SONET/SDH 信号速率对照表

SONET 信号 (STSs)	光纤载体 (OCs)	SDH 信号 (STMs)	速率 (Mb/s)
STS-1	OC-1	STM-0	51.84
STS-3	OC-3	STM-1	155.52
STS-9	OC-9	STM-3	466.56
STS-12	OC-12	STM-4	622.08
STS-18	OC-18	STM-6	933.12
STS-24	OC-24	STM-8	1 244.16
STS-36	OC-36	STM-12	1 866.24
STS-48	OC-48	STM-16	2 488.32
STS-96	OC-96	STM-32	4 976.64
STS-192	OC-192	STM-64	9 953.28

根据 SONET 的标准,SONET 的结构分为四个功能层,即物理层的光传输层,数据链路的段层、线路层、通道层,如图 1 所示。

ISO/OSI	SONET
数据链路层	通道层
	线路层
	段层
物理层	光传输层

图 1 SONET 的层次结构

(1) 光传输层 主要功能是提供电信号到光信号的转换,电 STS-n/STM-n 到光 OC-n 的映射,以提供高速率的光传输。

(2) 段层 主要处理 STS-n/STM-n 帧在光物理介质上的传输,包括形成帧、扰码、段差错控制和通信附加于帧上的段开销。

(3) 线路层 主要负责在物理线路上的传输,为通道层提供同步和复用、解复用功能。

(4) 通道层 主要负责处理通道终端设备之间的业务传送。

图 2 给出了一个由 SONET 的相关设备多路复用器(STS)、光再生器、多路复用与多路分解器

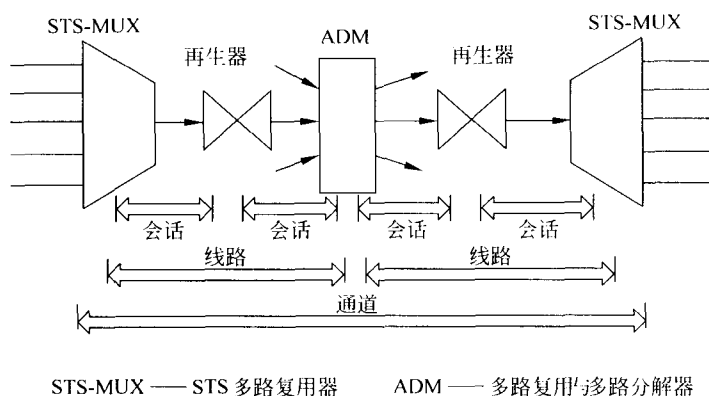


图2 SONET 系统

(ADM)、光纤所构成的同步光纤网系统示意图。

参考文献

1. Sidnie Feit. Wide Area High Speed Network. Macmillan Technical Publisher, 1999
2. 刘符, 韩焜国. 宽带通信原理、设计、应用. 北京: 人民邮电出版社, 1998
3. 韦乐平. 光同步传输网. 北京: 人民邮电出版社, 1993 (史美林)

tongyu

同余 (congruence) 两个整数 a 与 b 之差若是自然数 m 的倍数, 则称 a 与 b 模 m 同余。记为 $a \equiv b \pmod{m}$ 。这时 a 被 m 除所得的余数与 b 被 m 除所得的余数 (在 0 与 $m-1$ 之间) 相同。全体整数集合被分成 m 个互不相交的子集合, 每个子集合中的数模 m 互为同余, 不同子集合中的数模 m 不同余, 称这样的子集合为剩余类。从每个剩余类中取出一个数作为该剩余类的代表, 这 m 个数称为模 m 的一个完全剩余系。从两个剩余类中各取一个数, 这两个数之和 (差, 积) 所属的剩余类, 称为这两个剩余类之和 (差, 积)。在剩余类之间有加法、减法和乘法, 剩余类的集合成为一个环, 称为剩余类环, 记作 Z/mZ 。零所属的剩余类为该环的零元素, 1 所属的剩余类为该环中的乘法单位元。

一个剩余类中若有一个数与 m 互素, 则该剩余类中所有的数都与 m 互素, 称该剩余类与 m 互素。与 m 互素的剩余类的个数记为 $\varphi(m)$, 称 $\varphi(m)$ 为欧拉函数。设 $m = p_1^{l_1} p_2^{l_2} \cdots p_s^{l_s}$ 为标准因子分解式, p_1, \dots, p_s 为互不相同的素数, 则 $\varphi(m) = \varphi(p_1^{l_1}) \varphi(p_2^{l_2}) \cdots \varphi(p_s^{l_s})$, 即

$\varphi(m)$ 为积性函数, 而 $\varphi(p^l) = p^l - p^{l-1}$ 。从与 m 互素的每个剩余类中各取一数, 这 $\varphi(m)$ 个数组成一个模 m 的缩剩余类。若整数 a 与 m 互素, 则存在两个整数 x 与 y , 使 $ax + my = 1$, 从而 $ax \equiv 1 \pmod{m}$, 即 x 所属的剩余类是 a 所属的剩余类在乘法意义下的逆元素。所以与 m 互素的所有剩余类组成一个群。

当 $m = p$ 为素数时, 模 p 的剩余类之间可以有加、减、乘、除 (零元素不能作除数) 的运算, Z/pZ 是一个包含 p 个元素的域。

整数 a 若与 m 互素, 则 $a^{\varphi(m)} \equiv 1 \pmod{m}$ (费马小定理)。当 p 为素数时, 有 $(p-1)! \equiv -1 \pmod{p}$ (Wilson 定理, 其逆定理也成立)。

若 $f(x) = a_n x^n + \cdots + a_1 x + a_0$ 为整系数多项式, 求整数 a , 使其适合 $f(a) \equiv 0 \pmod{m}$, 称为求解同余方程。一次同余方程 $ax + b \equiv 0 \pmod{m}$ 有解的充分必要条件是 $(a, m) \mid b$ 。二次同余方程 $x^2 \equiv a \pmod{m}$ 若有解, 称 a 为模 m 的二次剩余, 否则称 a 为模 m 的二次非剩余。

参考文献

- 华罗庚. 数论导引. 北京: 科学出版社, 1957
(裴定一)

tongzhou dianlan

同轴电缆 (coaxial cable) 一种外带屏蔽层和保护皮以铜为导体的传输介质。同轴电缆共有四层, 如图 1 所示, 最内层的中心导体层主要成分是铜, 导体层的外层是绝缘层, 再向外为导体网, 最外层是表面的保护皮。

常见的同轴电缆线有:

- (1) RG-58A/U

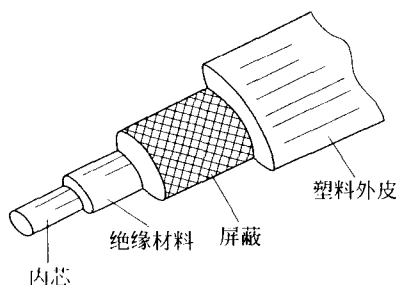


图1 同轴电缆的结构

用于10BASE2,阻抗为50欧姆,直径为0.18英寸,又称为“细同轴电缆”,是计算机网络中最常用的同轴电缆,以Ethernet标准而言,常需与BNC接头配合连接。

(2) RG-11

用于10BASE5,阻抗为50欧姆,直径为0.4英寸,又称为“粗同轴电缆”,需有收发器配合使用。

(3) RG-59U

阻抗为75欧姆,直径0.25英寸,常用于电视电缆。

(4) RG-62U

阻抗为93欧姆,直径0.25英寸,早期的ARC-net使用这种电缆。

同轴电缆所受的干扰较小,造价也相对较低,传输距离较长,这些是它的优点,缺点是:一根缆上的一点发生问题时,会影响到缆上所有的机器,影响整个网段的正常运行。

(王晓东)

tongyi jianmo yuyan

统一建模语言(unified modeling language, UML) 一种基于反映多种面向对象建模方法统一的可扩展建模语言。1997年统一建模语言(UML)被美国对象管理组织(OMG)采纳为该组织的建模语言规范。UML主要用于软件密集型系统的规约书写、系统构造、文档形成,并力求形象直观。它定义了建立系统模型所需的概念并给出其直观表示法,但并不涉及如何进行系统建模。因此,它只是一种建模语言,而不是一种建模方法。

UML的演化历史可概括为四个阶段。最初阶段是面向对象方法学家的联合行动,由G. Booch, J. Rumbaugh和I. Jacobson将他们各自的方法结合起来,形成统一方法0.8和0.9。第二阶段是公司的联合行动,有十多家公司组成UML伙伴组织,共同

提出UML1.0和UML1.1,作为向OMG提交的方案。第三阶段是在OMG的组织下进行修订和改进,产生了UML1.2、UML1.3和UML1.4等版本。目前所处的阶段是进行较为重大的修订。不久前已推出UML2.0版本。

基本成分

统一建模语言(UML)的基本成分是图与建模元素。UML定义了9种用于建立系统模型的图以及为各种图所公用的建模元素和扩展机制。图有类图、对象图、用案(或称用例)图、顺序图、协作图、状态图、活动图、构件图、部署图。其中类图和对对象图统称静态结构图;顺序图和协作图均为交互图;构件图和部署图均为实现图。建模元素有串、关键词、表达式、包、子系统等,并给出了其表示符号。扩展机制有约束、注释、标记值以及衍型。这些元素可以添加到其他建模元素之上,从而将原来的建模元素特化为一种语义较为特殊的新变形,或者表示出它们的某些细节。这样,将起到对UML扩充或细化的作用。

(1) 静态结构图——类图 and 对象图 UML的类图是诸如类、接口及其关系等静态说明的模型元素集合,用于展示模型之静态结构。和大部分面向对象建模方法不同的是,UML类图中作为结点的元素不只是类,也可以是接口、类型、包,甚至可以是实例。对象图是由实例构成的图形,包括对象和数据的值;UML称静态对象图为例图的实例,它显示了一个时间点上系统细节状态的一个快照。

(2) 用案图 用于表现参与者、用案及其间的关系。其中用案是由系统与一个或多个外部交互者(即参与者)之间交换的消息序列以及系统执行的活动共同体现的。参与者定义为一个实体的使用者在与该实体交互时所扮演的角色的集合。

(3) 顺序图 是UML提供的两种交互图之一,它展示按时间顺序排列的对象交互。特别是,它通过对象“生命线”来展示参加交互的对象,并按时间顺序展示它们之间所传送的激发。它所展示的交互是在一个协作中,对象之间为产生所要求的操作或结果而交换的一组信息。

(4) 协作图 是UML的另一种交互图,它表示一个协作,其中包含在一个特定上下文中由对象扮演的一组角色,以及它们所需之关系。协作图也可以表示交互,它定义了一组消息,这些消息说明了在协作中扮演角色的对象之间为达到所需的结果而进行的交互。和顺序图不同的是,协作图表现的是对

象角色之间的关系,并不表示时间顺序。

(5) 状态图 通过指明一个实体对于接收到的事件之反应而表现该实体的行为,特别是动态行为。

(6) 活动图 是状态图的变形,其中状态表示动作或子活动的执行,而转换是由动作或子活动的完成而触发的。它表现为一个过程本身的状态机。活动图用于对系统的行为建模。

(7) 实现图——构件图和部署图 是两种表现系统实现问题的图。其中构件图表现系统将实现的是哪些软件构件以及这些构件之间的依赖关系。构件是系统中一个符合一定标准、可部署、可替换、封装了实现而显露一组接口的部件,可部署到处理机结点上。部署图显示了运行时的处理单元以及在其上执行的软件构件、进程和对象所共同构成的配置。

新近发展

UML的出现使面向对象建模概念和表示法趋于统一和标准化。它应用广泛、成效显著,引起工业界和学术界的普遍重视。然而,UML也存在不足之处。例如,内容庞大,复杂无理,延拓或缺,严谨欠佳等。为此,OMG自1999年起即酝酿UML的下一重大发布,计划进行较大的实质性变动,以推出UML 2.0。经过数年的准备,UML 2.0已在不久前通过。其中虽有不少变动,但是否属实质性的变动,尚有待业者各抒己见。

参考文献

1. Fowler M, Scott K. UML Distilled: A Brief Guide to the Standard Object Modeling Language (2nd Edition). Addison-Wesley Longman Inc., 2000 (中译本: UML精粹(第二版),标准对象建模语言简明指南. 徐家福译. 清华大学出版社, 2002)
2. Kobryn Cris. UML 2001: A Standardization Odyssey Communication of the ACM. Oct. 1999, 42 (10)
3. OMG Unified Modeling Language Specification Version 1.4. Spet. 2001. <http://www.omg.org/cgi-bin/doc?formal/> (邵维忠)

tuling guiyue

图灵归约 (Turing reduction) 一种应用范围广泛的复杂性归约。其严格的形式定义要使用 oracle 图灵机。所谓 oracle 图灵机是一个多带图灵机 M , 它有一条特殊的工作带, 叫做 oracle 带或询问带, 和三个特殊的状态: 询问状态 $q_?$ 以及回答状态 q_y

和 q_N 。oracle 图灵机 M 的计算依赖于事先给定的 oracle 集 B 。当 M 处于询问状态 $q_?$ 时, 若 oracle 带上的字符串 $u \in B$, 则 M 转移到状态 q_y ; 若 $u \notin B$, 则 M 转移到状态 q_N 。无论是哪一种情况, 在这一步 oracle 带都被清成空白带, 而其他的带保持不变。这样一步计算称作询问。除询问之外, oracle 图灵机的计算和普通图灵机的计算相同。用 M^B 表示以 B 为 oracle 的 oracle 图灵机 M^B 接受的集合(语言)记作 $L(M^B)$ 。

设 A 和 B 是两个集合(语言)。如果存在 oracle 图灵机 M 使得 $A = L(M^B)$, 则称 A 可图灵归约到 B 。如果这个 M 是确定型多项式时间的 oracle 图灵机, 则称 A 可多项式时间图灵归约到 B 。 (张立昂)

tulingji

图灵机 (Turing machine) 英国数学家 A. M. Turing 于 1936 年提出的一种理想的计算机器的数学模型, 现在已成为计算机科学中可计算性理论和计算复杂性理论的基础。

图灵机分为确定型与非确定型两大类, 每类中又主要有单带、多带等许多形式与变种(已证明其计算能力是等价的)。一台标准的确定型单带图灵机由一条双向可无限长的被分为一个个小方格的磁带、一个有限状态控制器与一个读写磁头构成。图灵机一步步地进行工作, 机器工作情况取决于三点: 首先是机器的内部状态, 其次是读写磁头扫描在磁带的哪个方格上, 再次是读写磁头扫描着的方格上有什么信息。机器执行一步工作是如下进行的: 读写磁头在所扫描的方格上写上符号(原有的符号自然消除), 磁头向右或向左移动一个方格, 机器由当前所处的状态转向另一个状态, 然后进行下一步工作; 如此周而复始地机器一步步工作, 除非遇到某一命令机器停止工作的状态, 否则不停止。例如, 图 1 中的机器在某一步上处于状态 q_3 , 将所扫描方格中的 A 改写为 E , 左移一个方格, 进入状态 q_5 , 此时机器磁头扫描方格内的字母为 T ; 机器下一步的工作现在完全由状态 q_5 和扫描方格内的信息 T 惟一确定。

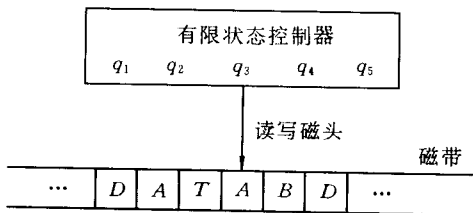


图 1 图灵机

图灵机的程序确定了图灵机的这种由状态、符号决定的一步一步的工作。有许多种方法来定义一台图灵机的程序(例如:流程图,伪汇编语言等),通常方便的方法是由下述五元组所确定的一个阵表来定义一台图灵机:

$$\langle q, A, E, M, q' \rangle$$

其中 q, q' 表示有限状态控制器中的状态; A, E 表示磁带方格上的符号, M 表示 L (左移)、 R (右移)或 N (不动);这样,上面例子中图灵机的一步动作可用五元组

$$\langle q_3, A, E, L, q_5 \rangle$$

来确切地描述。下述阵表是一个计算函数 $f(x) = 2^x$ 的图灵机程序,其中 B 代表磁带上的方格为空白方格,此外,一些约定如下:

- (1) x 和 $f(x)$ 的值以二进制表示;
- (2) 在开始时,磁带上只有一连续的方格串上放入相应于 x 的二进制值,其余方格均为空格;
- (3) 机器从状态 q_1 开始,磁头扫描在 x 最左位所在的方格上;
- (4) 停机时, $f(x)$ 的值就是磁带上非空格所组成的二进制串。

计算 $f(x) = 2^x$ 的图灵机程序如表 1 所示。

表 1 计算 $f(x) = 2^x$ 的图灵机程序

当前状态	B 被扫描时的 写、移动、 状态转移	0 被扫描时的 写、移动、 状态转移	1 被扫描时的 写、移动、 状态转移
q_1	$1, L, q_7$	$0, R, q_1$	$1, R, q_2$
q_2	B, R, q_3	$0, R, q_2$	$1, R, q_2$
q_3	$0, L, q_4$	$0, R, q_3$	Error
q_4	B, L, q_5	$0, L, q_4$	Error
q_5	Error	$1, L, q_5$	$0, L, q_6$
q_6	B, R, q_1	$0, L, q_6$	$1, L, q_6$
q_7	Halt	B, L, q_7	Error

表 1 中标有 Error 的地方表示在计算中不会出现,这个图灵机程序所确定的计算可用众所周知的图 2 所示流程图来解释,其中 $1 * y$ 表示在字符串 y 的左面添符号 1, $y * 0$ 表示在 y 的右面添加一个 0。

可以用更确切的方式将图灵机的程序定义如下:以 $Q = \{q_1, \dots, q_m\}$ 表示有限状态集, $\Sigma = \{a_1, \dots, a_n\}$ 表示磁带方格上的符号集,以 R, L, H 分别表示右移一格,左移一格或停机,一台确定型单带图灵机(的程序)由下述映射定义:

$$Q \times \Sigma \rightarrow \Sigma \times \{R, L, H\} \times Q$$

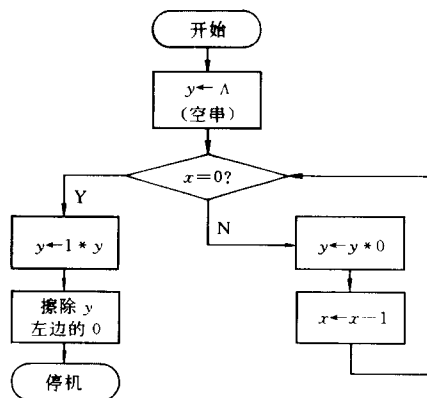


图 2 计算 $y = 2^x$ 的图灵机流程图

多带图灵机有一条输入带, k 条工作带;每条带上有一个读写磁头与有限状态控制器连接,它根据当前状态以及输入带和工作带上当前被扫描的符号决定下一步应该转到什么状态,应在工作带和输出带上写下什么符号以及每个磁头各自的移动方向是什么。非确定型图灵机与确定型图灵机的区别在于:机器从一个状态转到下一个状态时不是惟一的,它可以在两个或更多个状态中择取。

如果将字母表 Σ 上的一个字 $\alpha \in \Sigma^*$ 放在图灵机磁带上作为输入,图灵机 M 的磁头扫描在字 α 的开头字母上,并从状态 q_1 (定为初始状态)开始工作,则出现两种情况:要么在工作有限步后机器执行到某条停机指令停机,不再工作,这时磁带上输入的字 α 变成了输出的字 β ;要么机器永不停机,这时机器对输入的字 α 无输出。这样,一台图灵机 M 定义了一个从 Σ^* 到 Σ^* 的部分映射,称为图灵部分可计算映射,记为 ψ_M ;如果这个映射是处处有定义的,则称为图灵可计算映射。 Σ^* 的任一个子集合(语言) L 称为递归可枚举的,若有一台图灵机 M 使 $\text{dom } M = L$; L 称为是递归的(可判定的),若有一台图灵机 M 使得 ψ_M 恰为 L 的特征函数。判定问题是计算机科学理论研究中的一个重要问题。例如:已经证明,前后文有关文法的空虚性问题, Post 对应问题等都是不可判定的。在计算机科学中,证明许多问题的不可判定性都是采用划归到图灵机的停机问题上。可以如下叙述图灵机的停机问题:不存在一种算法来判定:对任意图灵机输入任意字 α 计算是否停机。

表面看来,图灵机的计算功能似乎很弱。但只要提供足够的时间(允许计算到足够多的步数)以及足够多的空间(允许使用足够长的磁带),则其力量

是非常强的,足以代替目前的任何计算机。图灵在设计了他的单带模型后提出:凡是可计算的函数都可以用一台图灵机来计算。这就是著名的图灵论题。A. Church 在提出 λ 演算时也说,可计算函数都是 λ 可计算的,这就是有名的丘奇论题。而后证明这是两个等价的命题,以后称之为图灵-丘奇论题。论题中使用了直观的非数学的“可计算函数”一语,因而论题本身是不能用数学方法来论证的,只能为实践所检验。大半个世纪以来,数学家、计算机科学家提出了各种各样的计算模型都被证明是同图灵机器等价的。这一论题已被当作公理,它不仅是计算机科学的基础,也是数学的基础之一。

图灵机器的计算对时(计算步数)空(磁带长度)是不加限制的;然而,现时世界对时空的限制却是必不可少的,如果对一个长为 1 000 的输入字要计算 2^{1000} 步,这在今日的大型计算机乃至可以想象的将来的计算机上都办不到。因此,自 20 世纪 60 年代起,对时空受限的图灵机器理论发展起来了,形成了今天的计算复杂性理论分支,出现了著名的尚未解决的所谓 P 与 NP 问题。P 是这样的语言类(即 Σ^* 的子集 L 组成的类): $L \in P$ 当且仅当存在一台确定型的图灵机 M 和一个多项式 f 使得对任何输入字 $\alpha \in \Sigma^*$, $\alpha \in L$ 当且仅当给图灵机 M 输入 α 时计算在 $f(|\alpha|)$ 步内停机并处于接受态,这里 $|\alpha|$ 是字 α 的长度;NP 是这样的语言类: $L \in NP$ 当且仅当存在一台非确定型的图灵机 M 和一个多项式 f 使得对任何输入字 $\alpha \in \Sigma^*$, $\alpha \in L$ 当且仅当给图灵机 M 输入 α 时计算在 $f(|\alpha|)$ 步内停机。P = NP 吗? 近 30 年来已发表了大量论文探讨它;数学、计算机科学中的许多重要问题都同它有关,例如现代公钥密码体系就建筑在 $P \neq NP$ 假定上。

参考文献

1. Kleene S C 著. 元数学导论. 莫绍译. 科学出版社, 1985
2. Rogers H. Theory of Recursive Functions and Effective Computability. New York: McGraw - Hill, 1967
3. 李祥. 可计算性理论导引, 贵州人民出版社, 1986 (李祥)

tulun

图论 (graph theory) 研究边和点的连接结构的数学理论。一个图 G 是一个三元组 $\langle V, E, \psi \rangle$, 其中 V 是一个非空有限集, 称为 G 的顶点集合, E 是有

限集, 称为 G 的边集合, ψ 是从边集 E 到顶点偶对集合的函数。图 G 的顶点数目称为它的阶。对任意的边 $e \in E$ 和顶点 $a, b \in V$, 若 $\psi(e) = (a, b)$ (无序偶对), 则称 e 为从 a 到 b 的无向边; 若 $\psi(e) = \langle a, b \rangle$ (有序偶对), 则称 e 为从 a 到 b 的有向边。这时, 称 a 为 e 的起点, b 为 e 的终点, 也称 e 关联于 a 和 b , 顶点 a 和 b 邻接。图中不与任何顶点邻接的顶点称为孤立点。所有顶点全为孤立点的图称为零图。关联于同一顶点的两条边称为邻接边, 关联于同一顶点的边称为自圈。对任意的边 $e_1, e_2 \in E$, 若 $\psi(e_1) = \psi(e_2)$, 则称 e_1 和 e_2 为平行边。不含自圈和平行边的图称为简单图, 有平行边的图称为多重图, 每一条边均为有向边的图称为有向图, 每一条边均为无向边的图称为无向图。既有无向边又有有向边的图称为混合图。若把有向图中每条有向边改为无向边, 则称所得到的无向图为该有向图的底图。路径是一个边的有穷序列 $e_1, e_2, \dots, e_n (n \geq 0)$, 其中 $e_i (1 \leq i < n)$ 的终点与 e_{i+1} 的起点相同。若 e_1 的起点和 e_n 的终点重合, 则称该路径为闭的, 否则称为开的。如果闭路径通过各顶点不超过一次, 则称它为回路。若从顶点 a 到 b 存在路径, 则称从 a 可达 b , 否则称从 a 不可达 b 。在无向图 G 中, 若任意两个顶点是可达的, 则称 G 为连通图。

设图 $G = \langle V, E, \psi \rangle$ 和 $G' = \langle V', E', \psi' \rangle$ 同为无向图或同为有向图。若 $V' \subseteq V, E' \subseteq E$ 且 $\psi' \subseteq \psi$, 则称 G' 为 G 的子图, 记为 $G' \subseteq G$ 。若 $G' \subseteq G$ 且 $G' \neq G$, 则称 G' 为 G 的真子图。若 $G' \subseteq G$ 且 $V' = V$, 则称 G' 为 G 的生成子图。

设图 $G = \langle V, E, \psi \rangle$ 且函数 $w: E \rightarrow R_+$ (R_+ 为非负实数集合), 则称 $\langle G, w \rangle$ 为加权图, w 为权函数, $w(e) (e \in E)$ 为 e 的权。如果 G 为有向图且其底图为连通的, 则称加权图 $\langle G, w \rangle$ 为网络, w 为容量函数, $w(e) (e \in E)$ 为 e 的容量。

E. Euler 在 1936 年解决了著名的哥尼斯堡七桥问题, 从而成了图论的创始人。七座桥将流经东普鲁士的哥尼斯堡市之普莱格尔河中的两个小岛及岛与河岸连接起来(见图 1)。问题是要从四块陆地 A, B, C 和 D 中任意一块出发, 经过每座桥恰好一

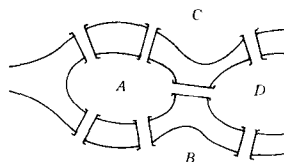


图 1 哥尼斯堡七桥示意图

次,再回到起点。人们经过多次试验都没有成功,这就是著名的七桥难题。最后 E. Euler 指出,七桥难题是无解的。他将陆地用顶点代替,桥用边代替,从而把七桥难题转化为图 2 所示的欧拉回路问题。

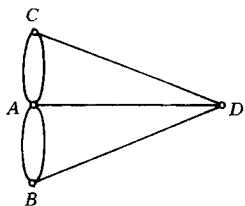


图 2 哥尼斯堡七桥模拟图

图论产生后,随即经历了一段半停滞时期。到了 19 世纪中叶,才又活跃起来,并在应用于化学和电路分析等方面,取得了重要成果。接着半个多世纪又几乎陷于停顿,直到 20 世纪中期,图论及其应用的研究才得到迅速发展,近 30 年来尤为显著。

到目前为止,图论所研究的基本内容有:图及其表示,有向图、无向图 and 多重图,树,平面图,二部图和网络等。图论所研究的典型问题有:

最短路径问题 它有两种类型,其一是求从加权图中任一给定顶点到其余各个顶点的最短路径,其二是求加权图中任意两给定顶点间的最短路径。这里的“最短”是指路径的加权长度最短。

着色问题 在对一个图的顶点进行着色时,为了保证相邻顶点的着色均不相同,至少需要多少种颜色? 已经证明 5 种颜色是足够的,3 种颜色是不行的。人们猜测 4 种颜色最合适,这就是著名的四色问题。

环球旅行问题 威廉·哈密顿 1957 年发明了一种游戏,就是在实心正十二面体的 20 个顶点处标以有名的城市名字,要求游戏者找一条沿着各边通过每个顶点恰好一次的闭路径(称为回路),即“环球旅行”。由此引出了哈密顿回路和哈密顿图等概念。迄今为止,尚未获得图中存在哈密顿回路的充要条件。

欧拉路径问题 通过一个图中每条边恰好一次的路径称为该图的欧拉路径,闭欧拉路径称为欧拉回路。一个图中是否存在欧拉路径,以及如何求出存在的欧拉路径,就是所谓的欧拉路径问题。它已由 E. Euler 彻底解决了。

平面图问题 在现实生活中,常常要画一些图形,并希望边与边之间尽量减少相交的情况,例如印刷线路板上的布线,交通道路的设计等。那么,一个

图能否画在一个平面上,使其边不在非顶点处交叉呢? 由此引出了平面图、对偶图等概念。

极小连通问题 有 n 个城市 A_1, A_2, \dots, A_n , 在这些城市之间要建造一个公路系统,使得旅行者从任何一个城市可以到达其他所有城市。那么,最少要修建多少条公路,才能构成上述公路系统? 怎样使总的费用达到最小? 由此引出了树、生成树、最小生成树等概念。

完美匹配问题 有 n 个人 x_1, x_2, \dots, x_n 和 m 台机器 y_1, y_2, \dots, y_m 。若 $x_i (1 \leq i \leq n)$ 可以操作机器 $y_j (1 \leq j \leq m)$, 就在 x_i 和 y_j 之间连一条边。在工作过程中,每个人只能操作一台机器,且每台机器也只能由一个人操作。试问:应如何适当地安排,才能使尽可能多的人工作。由此引出了二部图、极大匹配、完美匹配等概念。

图的矩阵表示问题 为了有效地利用计算机对图进行处理。首先,必须给出一个图在机器内部的存储形式。为此,引出了图的邻接矩阵、可达矩阵、关联矩阵等。通过图的矩阵表示,也使我们能用代数工具研究图的一些性质。

网络流问题 假设把某种物资从产地 A 运到销售地 B , 途经若干中转站。每两个中转站之间的货运量都有一个最大限量(称为该段道路的容量),如何制定从 A 到 B 之运输量最大的方案,就是所谓网络的最大流问题。

图论是一门应用性很强的学科。应用图论来解决运筹学、化学、生物学、网络理论、信息论、控制论、博弈论和计算机科学的问题,已显示出极大的优越性。

(张强)

tulun suanfa

图论算法 (graph algorithm) 关于图论问题的求解算法。图是计算机科学中一种常见的数据结构,因而与图有关的算法在计算机科学中也十分重要。采用图来描述的有意义的计算问题成百上千。

一般来说,图在计算机中的表示方法有邻接矩阵表示法、二进制位向量表示法、邻接表表示法、邻接向量表示法和关联矩阵表示法。对于一些特殊的图,还有一些特殊的表示方法。总的来说,各种表示方法各有其优缺点,采用不同的表示方法,可获得不同的时空性能。对于具体的问题,应根据对图要进行哪些操作,按什么方式操作以及各种操作的使用频率等因素来决定采用何种表示方法,以获得时空性能令人满意的算法。

搜索一个图是指从某点出发按某种方式沿着该图的边前进,以访问该图的所有顶点。图的搜索问题是图论中的基本问题,许多图论算法以图的搜索算法为基础。宽度优先搜索和深度优先搜索是两种基本的搜索图的算法。

给定图 $G=(V,E)$ 和源 $s \in V$, 宽度优先搜索只有在发现了与 s 相距 k 的所有顶点之后,才开始发现任何与 s 相距 $k+1$ 的顶点,这里两点之间的距离是两点之间“最短路径”的长度,即其中包含的边数。深度优先搜索遵循的策略是尽可能在图中搜索更深的顶点。其中,对最近发现的顶点 v ,若有尚未考察的边离开 v ,则考察之。在考察完所有这些边后,搜索将回溯去考察离开顶点 u 的边,这里顶点 v 是从顶点 u 发现的。深度优先搜索可用于形成有向无圈图的拓扑序,还可用于将有向图分解为强连通分支。

在最短路径问题中,给定一个加权的有向图 $G=(V,E)$,其权函数 $w: E \rightarrow \mathbf{R}$ 把边映射成实数权值。路径 $p = \langle v_0, v_1, \dots, v_k \rangle$ 的权是其中各边的权之和 $w(p) = \sum_{i=1}^k w(v_{i-1}, v_i)$ 。从 u 到 v 的最短路径 $\delta(u, v)$ 定义为:若存在从 u 到 v 的路径,则它是从 u 到 v 的所有路径的权的最小值,否则它为 ∞ 。从顶点 u 到顶点 v 的最短路径定义为具有权 $w(p) = \delta(u, v)$ 的任何路径 p 。

在单源最短路径问题中,给定一个图 $G=(V,E)$,希望找出从某个给定源 $s \in V$ 到每个顶点 $v \in V$ 的最短路径。许多其他问题可用单源问题的算法来求解。单源最短路径问题求解算法中使用的主要技术是松弛,这种技术反复地减少每个顶点的实际最短路径权的上界,直到该上界等于最短路径权时为止。各种求解算法之间的差别在于它们松弛每条边的次数以及松弛边的顺序。在 Dijkstra 算法和有向无圈图的最短路径算法中,每条边刚好松弛一次。而在 Bellman-Ford 算法中,每条边要松弛多次。Bellman-Ford 算法虽然比 Dijkstra 算法更通用,但执行时间却比后者多。按照加权有向无圈图中顶点的拓扑序来松弛离开各顶点的所有边,可快速地计算单源最短路径。单源最短路径问题求解算法可用于求解差分约束系统。

在每对顶点间最短路径问题中,给定一个图 $G=(V,E)$,希望找出每对顶点 $u, v \in V$ 之间的最短路径。虽然可依次以每个顶点作为源,反复运行单源最短路径算法 $|V|$ 次来求解这个问题,但是还有一些更好的求解算法。一般来说,可采用动态规划

的方法来求解这个问题。性能较好的求解每对顶点间最短路径问题的 Floyd-Warshall 算法也采用了动态规划的方法。Floyd-Warshall 算法的思想也可用于求有向图的传递闭包,在支持位向量运算的计算机上,该算法既省空间又省时间。求解每对顶点间最短路径问题的 Johnson 算法对于边较少的所谓稀疏图,其性能较好。该算法采用重置权技术,把原问题变换成另一个等价的问题来求解。

事实上,称为“闭半环”的代数结构可为求解有向图的路径问题产生一种一般的框架。许多算法都可看成闭半环上用于计算每对顶点间路径信息的一般算法的实例。

有关计算一个图的最小权生成树的算法参见最小生成树。有关计算一个有向图的最大流的算法参见最大流。

参考文献

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. Introduction to Algorithms. The MIT Press, 1990
(殷建平 陈火旺)

tuxiang bianjie biaoshi

图像边界表示 (image boundary representation) 把图像中的物体轮廓或边界用合适的数据结构或数字串表示的方法。图像边界表示的方法可分为链码表示、折线近似表示、 ψ -S 曲线表示以及傅里叶描述子表示等。

链码表示 图像的边界是由一串离散的像素点组成的,如果将图像用一网格覆盖,使得像素点位于网格的交点上,那么边界可以看成是由一系列短线段组成的链,其中每个短线段正好是网格相邻交点的连线。

链码表示是将相邻短线段的方向向量进行编码。有 4 链码和 8 链码之分。8 链码有 8 个方向,分别用 0,1,2,3,4,5,6,7 共 8 个数分别表示 $0^\circ, 45^\circ, 90^\circ, 135^\circ, 180^\circ, 225^\circ, 270^\circ$ 和 315° 的 8 个方向(见图 1)。链码就是从起始点开始,沿曲线观察每一线段的走向,并用相应的方向码来表示,例如在图 2 中的曲线

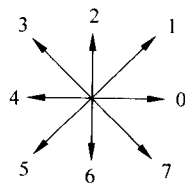


图 1 8 链码的 8 个方向

之链码为:0 1 2 2 3 2 2 1 0 0 0 0 7 6 5 5 6 7 1 1。

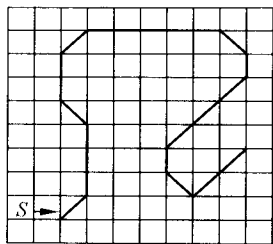


图2 用链码表示曲线

为了确定链码所表示的曲线在图像中的位置,并由链码准确地重建曲线,必须标出起始点的坐标。但有时并不关心起始点的具体位置,特别是当曲线为闭合时,因为起始点位置的变化只引起链码位移的缘故。此时,可以通过改变起始点位置使表示起始点坐标的整数最小,这个过程称为链码的规格化。

折线近似表示 用许多折线段的组合来近似表示曲线边界的方法。其基本做法是将边界不断地用折线段划分下去,直到满足某一结束条件为止。具体方法如下:

- (1) 将边界两端点用直线连接起来。
- (2) 计算边界上的各点到该直线的距离,若其中最大距离小于给定阈值,结束。
- (3) 将最大距离所对应的边界点作为划分点。
- (4) 去掉连接端点的直线,将两端点分别与划分点用直线连接,转至(1)。

上述过程可用图3表示。

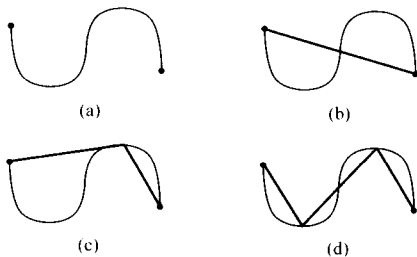


图3 曲线的折线近似表示

ψ -S 曲线表示 这种方法是通过测量边界上各点处的切线与 x 轴的夹角 ψ 和相对于某一起始点各切点的弧长 S 来表示边界的形状。

傅里叶描述子表示 这是一种闭合边界的常用表示方法。

任何闭合边界可以在 xy 复平面上表示为

$$u(n) = x(n) + jy(n) \quad n = 0, 1, \dots, N-1$$

这是一个以 N 为周期的周期函数,其傅里叶变换表示为

$$u(n) = \frac{1}{N} \sum_{k=0}^{N-1} \alpha(k) \exp\left(\frac{j2\pi kn}{N}\right)$$

$$0 \leq n \leq N-1$$

其中复系数 $\alpha(k)$ 为

$$\alpha(k) = \frac{1}{N} \sum_{n=0}^{N-1} u(n) \exp\left(\frac{-j2\pi kn}{N}\right)$$

$$0 \leq k \leq N-1$$

这里称 $\alpha(k)$ 为边界的傅里叶描述子。它具有如下特点: ①边界的许多几何变形,如平移、尺度变化、旋转等都可以通过傅里叶描述子的简单运算得到; ②用傅里叶描述子可以重构原边界; ③利用傅里叶描述子可以将不同大小、不同朝向的类似形状加以匹配。

除上述几种的方法之外,还有二次、三次曲线等边界的表示方法。

参考文献

1. 赵荣椿等. 数字图像处理导论(第二版). 西安: 西北工业大学出版社, 1995
2. Jain A K. Fundamentals of Digital Image Processing. Prentice-Hall Inc., 1989
3. Gonzales R, Woods R. Digital Image Processing. Addison-Wesley Publishing Company Inc. 1992

(曾建超)

tuxiang bianyuan jiance

图像边缘检测 (image edge detection) 检测或确定图像中不连续特性的技术。图像中的不连续往往有一定的意义,通常代表了不同物体的边界或者同一物体不同表面的分界。

边缘检测对于**图像分割**与理解有重要意义。图像分割可以根据确定的边界来进行。边界的形状提供了关于物体形状的信息,对于识别图像中的物体有相当作用。边界的走向及重叠等信息对图像中物体的结构(相对位置)的理解也有相当意义。

图像中的边缘是图像中两个区域的边界,而这两个区域在亮度上(或颜色上)有相当差别。通常认为,边界处的亮度有较突然的变化。从数学的角度看,边界处的亮度对空间(二维图像平面坐标)的导数有较大的绝对值。所以,大部分的边缘检测技术都基于求导数的方法。

由于计算机并不是用连续的函数处理大部分图

像,而是用一些离散点上的值来表示。所以求导数的算法也必须用离散的差分算法(在图像处理中,差分是用一些提取特征的算子来表示,可参见**图像特征提取**中有关内容)。

在边缘检测中,经常用到亮度梯度的概念。图像亮度梯度是一个向量,其分量是图像亮度沿某两个坐标方向求偏导数所获得的值。图像 $f(x,y)$ 看成一个连续函数可以表示为

$$G = \begin{bmatrix} G_x \\ G_y \end{bmatrix}, G_x = \frac{\partial f(x,y)}{\partial x}, G_y = \frac{\partial f(x,y)}{\partial y}$$

在实际的数字图像上,梯度的两个分量可以用下面的模板算子来计算

$$\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \quad \text{和} \quad \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$$

梯度的幅度是

$$\sqrt{G_x^2 + G_y^2}$$

梯度的方向是

$$\tan^{-1}(G_y/G_x)$$

上面提及的算法以及在图像特征提取中的算法都是利用较小邻域的信息(参见**图像邻域运算**)对孤立点来进行的。通常在采用这些算法之后,还要作进一步的检测及连接工作,以获得有意义的物体的边界或者物体表面的分界线。为了进一步确认及连接边界,一般可采用两种方法:局部分析法和全局分析法。

(1) 局部分析法 是最简单的一种确定连接的方法。考虑一个小的邻域,例如 3×3 或 5×5 的邻域,在这个邻域里,把相似的点连接起来形成具有某种共同的特性的边界。通常,相似性要考虑两方面:一是用边界检测算子(梯度)计算后获得的强度(梯度的幅度);二是导数值沿两个坐标方向的比例(梯度的方向)。在某个图像区域内,如果检测出来的两点在梯度幅度及方向上的差别都小于某一给定值,就认为它们属于同一条边界,可以在图像中把它们标记出来。这个过程可以对整个图像分块地重复进行。

(2) 全局分析法 采用全局分析法时,要研究被检测出来的可能是边界点的像素在整个图像平面中的相互关系。霍夫变换(参见**图像特征提取**)可以用来进行全局性的分析确定。

参考文献

1. 赵荣椿等. 数字图像处理导论(第二版). 西安:西北工业大学出版社,1995

2. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989 (俞志和)

tuxiang bianhuan

图像变换 (image transformation) 改变图像表示方法的技术。图像变换通常可以使图像处理更方便、有效。图像变换在**图像增强**、**图像复原**、**图像压缩**(减少表示图像的数据)以及**图像特征提取**等方面都有十分重要的应用。

通常,图像变换应遵循以下原则:

(1) 变换应是可逆的,也就是说,经过变换的图像还应该可以变换回来。

(2) 变换应有利于进一步的运算或处理。

(3) 变换的算法不应该太复杂。

最常用的图像变换方法是二维的傅里叶变换。它将二维的空间量(图像亮度值),即我们平常看到的图像,用频率函数,或者说,用一系列不同频率的正弦函数来表示。通常,人们把傅里叶变换称为从空间域到频率域的变换。其他一些常用的变换方法有 Walsh、Hadamard、cosine 和 Hotelling 也即 K-L (Karhunen-Loeve) 变换等。

图像的傅里叶变换 一幅图像可以用一个二维空间函数 $f(x,y)$ 来表示。这里, x 和 y 可以认为是沿着图像水平方向及垂直方向的距离(坐标)。如果图像函数 $f(x,y)$ 满足在 $-\infty$ 到 $+\infty$ 的范围内绝对可积的条件,那么,我们就可以定义下面的积分

$$F(u,v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x,y) \exp[-j2\pi(ux + vy)] dx dy$$

为二维函数 $f(x,y)$ 的傅里叶变换。这里, \exp 表示自然对数的底, j 是虚数单位。

$F(u,v)$ 可以认为是一个二维频率函数,而 u,v 分别是沿着水平及垂直方向的频率。例如,一个小圆点的图像,如图1(a)所示,经过傅里叶变换后,其频率函数的幅值,可以用图1(b)来表示。图1(b)的中间部分很亮,表示低频部分的幅值较大。围绕中心的亮环表示有一些较大幅值的中等频率的分量,而比较黑的最外面部分表示高频部分的幅值较小。

图2(a)是一个长方形,经过傅里叶变换后,它的频率函数的幅值表示在图2(b)中。因为该长方形沿水平方向比垂直方向窄,傅里叶变换后,沿水平方向的大幅值的低频部分比垂直方向的低频部分反而宽。这个现象体现了傅里叶变换的方向特性。

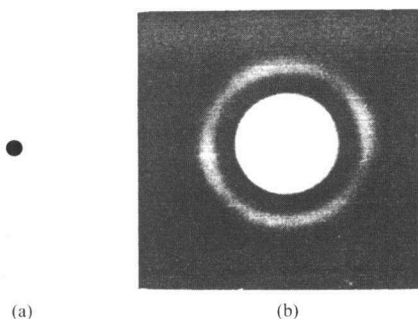


图1 小圆点(a)及其频率函数的幅值的图像(b)
(此图取自参考文献3)

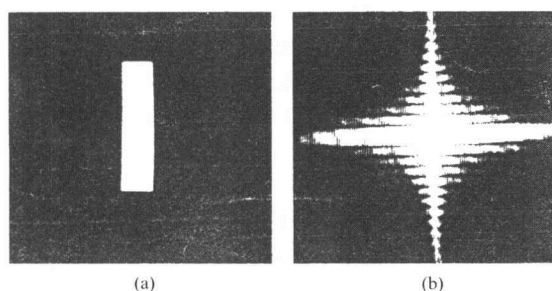


图2 长方形(a)及其频率函数的幅值的图像(b)
(此图取自参考文献3)

因为变换的目的是要有利于进一步的运算或处理(参见图像变换运算)。那么经过在频率域中的运算或处理后,通常需要进行傅里叶反变换,或者说,把频率函数变换回空间函数。傅里叶反变换的表达式是:

$$f(x, y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} F(u, v) \exp[j2\pi(ux + vy)] du dv$$

离散傅里叶变换 在一般情况下,计算机里的图像并不是用二维连续函数来表示,而是用一些离散点上的数值来表示的(参见图像模型)。所以,在大部分的情况下,傅里叶变换要用这些离散点上的数值来进行,这就是离散傅里叶变换。常用缩写DFT来表示。

假定一幅图像沿水平方向有 M 个点,沿垂直方向有 N 个点。图像在某一点的灰度或亮度值用 $f(m, n)$ 来表示。这里, m, n 表示像素点的位置。那么,二维离散傅里叶变换(DFT)可以表示为:

$$F(k, l) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) \times \exp[-j2\pi(mk/M + nl/N)] \quad (1)$$

其反变换为

$$f(m, n) = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} F(k, l) \times \exp[j2\pi(mk/M + nl/N)]$$

二维DFT具有某些有用的或方便使用的性质,其中包括:

(1) 可分离性 二维DFT可先沿某一方向(如水平方向)做一维的DFT变换,将结果再沿另一方向(如垂直方向)做一维DFT。得到的变换结果 $F(k, l)$ 与一维DFT的次序无关。

(2) 线性 两幅或多幅图像的线性组合的DFT等于它们分别变换的线性组合。假如用 J 表示DFT的变换过程公式(1),那么线性性质可表示为:

$$\begin{aligned} J[af_1(m, n) + bf_2(m, n)] \\ = aJ[f_1(m, n)] + bJ[f_2(m, n)] \end{aligned}$$

即,图像 f_1 乘上常数 a 加上图像 f_2 乘上常数 b 的和的变换等于 f_1 的变换乘上一个常数 a 加上 f_2 的变换乘上常数 b 的和。

(3) 比例关系可以用下式来表示

$$J[f(am, bn)] = \frac{1}{|ab|} F(k/a, l/b)$$

这就是说,如果原图像在水平方向放大(或缩小) a 倍,在垂直方向放大(或缩小) b 倍,那么,变换后的函数在水平方向的频率缩小(或放大) a 倍,在垂直方向频率缩小(或放大) b 倍,而且变换后的幅值缩小(或放大) $|ab|$ 倍。这里 $|ab|$ 表示 ab 的绝对值。

(4) 空间平移性质可以表示为

$$\begin{aligned} J[f(m - x_0, n - y_0)] \\ = F(k, l) \exp[-j2\pi(kx_0/M + ly_0/N)] \end{aligned}$$

可以看到,当空间图像 f 沿水平方向移动了 x_0 ,沿垂直方向移动了 y_0 以后,它的傅里叶变换中各种频率分量的幅值不变,只是相位有所变化。式中右边的 $\exp[-j2\pi(kx_0/M + ly_0/N)]$ 只是表示相位的变化。这种被称为“平移不变”的性质很有用,常常被用在图像特征提取或图像中的模式识别等。

(5) 周期性用公式表示是

$$F(k, l) = F(k + pM, l + qN)$$

这里 p, q 可以是任意正整数,即 $1, 2, 3, \dots, M$ 是空间图像沿水平方向的像素点数,而 N 是图像沿垂直方向像素点数。这就是说,在频率域中,图像的DFT沿水平方向每隔 M 个频率(采样)点重复一次,沿垂直方向每隔 N 个频率(采样)点重复一次。

(6) 平均值可以表示为

$$F(0, 0) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n)$$

这就是说,频率函数的零频率分量等于空间图像亮度(灰度)的平均值。

(7) 相关 如果有两幅图像, $f(m, n)$ 及 $g(m, n)$ 。它们都是用实数来表示的(注意,图像也可能用复数来表示,例如在频率域中,前面的一些表达式中有虚数 j),那么,它们的相关函数是这样的

$$f(m, n) \circ g(m, n) = \sum_{\alpha=0}^{M-1} \sum_{\beta=0}^{N-1} f(\alpha, \beta) g(m+\alpha, n+\beta)$$

m 可以取值 $0, 1, 2, \dots, M-1$, n 可以取值 $0, 1, 2, \dots, N-1$ 。假如取 $m=2, n=3$,那么,上面相关函数的值就是图像 f 与水平方向平移了 2 个像素及垂直方向平移了 3 个像素的图像 g 的各相应像素值乘积之和。

DFT 的相关性质是:

$$J[f(m, n) \circ g(m, n)] = F^*(k, l) G(k, l)$$

这就是说,通过 DFT,相关的计算变成了乘法运算。这里 $F^*(k, l)$ 表示 $F(k, l)$ 的共轭函数。

相关计算的一个重要应用是进行“模板匹配”。例如,用一幅图像作为参考图像,我们要在其他一些图像中寻找与参考图像最相近(相像)的图像,那么相关值最大的两幅图像可以被认为最“像”。利用傅里叶变换来进行相关计算往往效率较高。

从离散傅里叶变换的公式中可以发现,计算中的加法与乘法次数是与图像中的像素点的平方成正比的。换句话说,当图像尺寸增加时,计算量会急剧增加。利用一维快速傅里叶变换和二维 DFT 的可分离性可以把计算次数减少。

参考文献

1. 余松煜,周源华,吴时光. 数字图像处理. 北京: 电子工业出版社, 1989
2. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989
3. Rafael C, Gonzalez/Paul Wintz. Digital Image Processing (second edition). Addison - Wesley, 1987
(俞志和)

tuxiang bianhuan yunsuan

图像变换运算(image transform operation)

供处理图像变换用的运算。

变换运算的基本过程是,先将原图像 $f(x, y)$ 经过正交变换(例如离散的傅里叶变换)转换到频域 $F(u, v)$ 。然后,在频域中,对经过变换的图像进行运算得到频域中滤波的结果 $G(u, v)$ 。最后,再将结果进行反变换,就得到滤波后的图像 $g(x, y)$ 。常用的

变换是傅里叶变换。当从频域来研究图像时(参看**图像变换**),图像傅里叶变换结果中的直流分量对应于图像的平均灰度,低频分量对应于图像中缓慢变化的部分,而高频分量对应于图像的边缘、细节和噪声。因此低通滤波(让低频部分通过)可用于图像平滑或去除噪声,高通滤波可用于图像锐化和边缘提取(参见**图像邻域运算**),而带通滤波可能在增强图像边缘和其他高频成分的同时又可以抑制噪声。

频域中的线性滤波

$$G(u, v) = F(u, v) H(u, v)$$

对应于空域中的线性滤波(卷积)

$$g(x, y) = f(x, y) * h(x, y)$$

这里 $H(u, v)$ 是 $h(x, y)$ 的傅里叶变换,称为线性滤波器传递函数。 $h(x, y)$ 是点扩展函数。这个性质称为卷积定理。它说明空域中的卷积运算可转换为频域中的乘积运算。

参考文献

1. 余松煜,周源华,吴时光. 数字图像处理. 北京: 电子工业出版社, 1989
2. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989
(朱志刚)

tuxiang biaooshi

图像表示(image representation) 将图像或图像中的某些部分(例如物体的轮廓)或某些区域用合适的数据结构或数学式子表示的方法。

一幅图像通过分割和特征提取之后,往往被分成一些不同特征的部分。为了利用计算机对图像进行更高层次的处理(例如,进行图像景物理解、物体识别以及对图像进行各种描述等),需要将这些特征的部分用合适的数据结构或数学式子表示出来。例如,通过边界提取处理之后(参见**图像边缘检测**),图像中相互关联的边界被连接起来,它可以大致说明图像中物体的形状。但这只是在像素层次上的说明,是通过人来理解的,而计算机本身还并不容易判别出物体的确切形状。在轮廓提取的基础上,如果利用链码、傅里叶算子等方法将轮廓明确地表示出来,那么计算机就可以方便地判别出物体形状,进而达到检测、识别景物中的物体或利用**计算机辅助设计(CAD)**技术来合成物体形状的目的。

图像表示的几种常用方法是:轮廓表示方法、区域表示方法、几何特征表示方法、矩表示方法以及骨

架结构表示方法(参见**图像轮廓表示**、**图像区域表示**、**图像几何特征表示**、**图像矩表示**以及**图像骨架表示**)。(曾建超)

tuxiang binxing chuli

图像并行处理(image parallel processing)

并行处理图像的过程和技术。对于绝大多数图像处理过程,适当的计算支持极其重要。即使简单的图像处理问题,计算的要求也相当高。视频图像的典型分辨率是每帧 $512 \times 512 \times 24$ 位,而视频速度实时处理意味着每秒需处理25或30帧。如果对整幅图像进行 3×3 的实时卷积,则要求处理速度高达8 000万次/s的八位乘法和加法,此外还要加上输入和输出等开销(注意,卷积只是图像处理中最基本的运算之一)。所幸的是,图像处理中的许多算法尤其是低层处理,对图像中的所有像素执行相同的操作,这就提供了图像处理并行处理的可行性。所以,研究适合于图像处理的并行结构(参见**并行处理系统**)和并行算法是十分需要也是可行的。

图像并行处理的层次 图像处理和分析的全过程可分为低层图像处理(图像预处理)、中层图像处理(图像分割和描述)和高层图像处理(图像识别和解释)。在从低层到高层的图像处理过程中,数据表示由简单到复杂,操作由局部到全局。因此,应该根据不同层次图像处理任务的特点选择最适合的计算机体系结构及算法。

低层图像处理 主要指图像的预处理和滤波。包括**图像增强和滤波**(参见**图像处理的基本运算**)、**图像复原**、**图像变换**、**图像边缘检测**、**形态学滤波**(参见**图像形态学运算**)等。低层图像处理的主要特点是:①输入和输出有相同的数据结构,即二维图像点阵;②对所有的像素执行相同的操作;③像素之间的联系限于较小的邻域范围,即对像素施加的是局部操作。因此与之相适应的并行体系结构是单指令流多数据流(SIMD)结构,其主要类型是高度并行的阵列处理机。由于低层图像处理具有统一的数据结构,所以也适于采用流水线结构。

中层图像处理 是介于低层和高层图像处理之间的桥梁。一般进行由点阵图像数据到符号数据的变换。中层图像处理的输入是点阵图像,输出是符号数据。其处理主要的内容是图像分割和描述,包括像素分类、区域分割、霍夫变换(参见**图像特征提取**)、细化、骨架提取和各种图像表示等(参见**图像表示**)。由于输出数据结构的多样性,设计适合于

中层图像处理的最佳并行体系结构相当困难。与低层处理相比,中层处理有以下几个特点:①由于处理中仅仅关心若干特征点,并行处理单元的数量可以减少,这样并行的程度也相应降低了;②对不同特征的处理方法可能不同,所以要求提高处理单元的自主能力;③中层处理不仅包括局部操作,也包括全局操作,所以要求较大范围内的处理单元能直接通信,因此,需要更复杂的互连网络。

高层图像处理 是对图像的理解,包括对图像中物体的识别及定位。高层图像处理的对象主要是符号数据,这些数据涉及图像的全局信息,执行的操作也是全局性的。高层处理有3个特点:①任务中包含大量的测试和转移;②数据结构不规则,从而对数据的检索和处理依赖于数据本身;③数据的复杂性比算法的复杂性更影响处理时间。这些特点决定了高层处理应当开发进程级、任务级并行性,要求各个处理机能够自主地执行自己的程序。所以对高层图像处理来讲,较好的并行体系结构是多指令流多数据流的结构。

另外,在一个图像处理系统中,不同层次的结构之间必须有良好的通信机制,以高效地完成数据和控制的层间传递。

图像并行处理机制 有3种类型的并行处理机制。它们是数据并行处理、流水线并行处理及任务并行处理。

(1) 数据并行处理将数据分布到各处理机上,并在所有的处理机上执行相同的代码。数据并行的计算机映射到单指令流多数据流结构的编程模型上,各个处理机在自己的局部数据上同步执行相同的代码。数据并行程序设计的优点是控制流简单,数据阵列简单地从一个状态变为另一个状态,处理结果是确定的,并和系统中物理处理机的数目无关。这就意味着数据并行程序调试简单,移植方便。但是数据并行算法较适合于低层图像处理,难以实现高层图像推理。如果处理机的数目少于图像中像素的个数,则可以将图像分块处理。

(2) 流水线并行处理指脉动式或流水线模型,将算法在时间上划分成流水段建立起流水线,每段计算出部分结果后传到下一段。许多单指令流多数据流的算法可在脉动阵列上实现。但不同的是,在流水线上,处理机是在不同的时间执行相同的操作。

(3) 任务并行处理是将子任务分布到各个处理机上。在多机系统上编程远比在单处理机上编程困

难,因为对多机系统有如下要求:①任务需要分解;②子任务和数据需分配到各处理机上;③建立处理机之间的通信和同步机制。这3个步骤必须以一定的顺序完成,因为通信和同步不能在程序分配之前决定。对于新的算法和结构,上述过程必须重新确定。在消息传递的超立方体多指令流多数据流机上,处理机之间通信的代价会很高,所以任务的分布要使得通信量最小。

图像并行处理的运算 所有图像处理的运算都可通过利用适当的邻域运算(参见**图像邻域运算**)序列来实现。这些邻域运算可分为三类:

(1) 点运算 在某个像素处的结果只是该像素原来值的函数,从而在邻域像素之间不需要传递数据。此为邻域运算的下限情况。

(2) 局部邻域运算 某个像素的结果是局部邻域(如 3×3)中诸像素值的函数。 3×3 邻域运算更为常用,它容易在局部连接的单指令流多数据流计算机上实现。

(3) 全局邻域运算 送到邻域处理机的数据是本点数据和从邻域处理机上所获得的数据的函数。在这种方式下,单个指令可以将信息传播到阵列中任意远的距离,因此某像素的结果可以是整个阵列中数据的函数。全局操作是邻域运算的上限,其特殊情况是邻域窗口为整幅图像。

有些通用的算法在图像并行处理的过程中都非常有用,如矩阵相乘、排序和搜索、离散傅里叶变换、动态规划和松弛算法等。

参考文献

Hussain Z. Digital Image Processing: Practical

Applications of Parallel Processing Techniques. Ellis Horwood Limited, 1991

(朱志刚)

tuxiang chongjian

图像重建 (image reconstruction) 根据被测对象的投影数据重建被测对象图像的技术。获得对象投影数据的方法有放射法(对象本身放射能量)、反射法(对象反射电子束光或超声波等)、透射法(X射线透过对象)。这里以医学放射学方面的应用为例只介绍图像重建的透射法。

根据人体断面的投影重建此断面的图像,在医学上称作计算机断层摄影技术,简称CT,是医疗诊断的重要手段,已发展成医学图像处理的专门学科分支。

CT扫描仪就是利用X射线透射方法重建图像。它有两个主要部分:包括放射源和传感器的机架以及信号处理单元。放射元件及量测元件通常固定在机架上。对于小型的扫描仪,量测部分可以用手持。较大型的断层扫描仪,其机架可以旋转。信号处理单元包括计算机及显示设备。

放射源运动方式原则上分为两类:平行扫描和扇形扫描,见图1。也有用其他方式的,但都是平行扫描和扇形扫描的组合。用平行扫描完成一个断面的量测,一般需要几分钟。扇形扫描可以把扫描时间缩短到几秒到几十秒,但量测元件的数目将增加很多(如图1(b)作为检测元件的上千个碘化钠晶体分布在圆周上)。

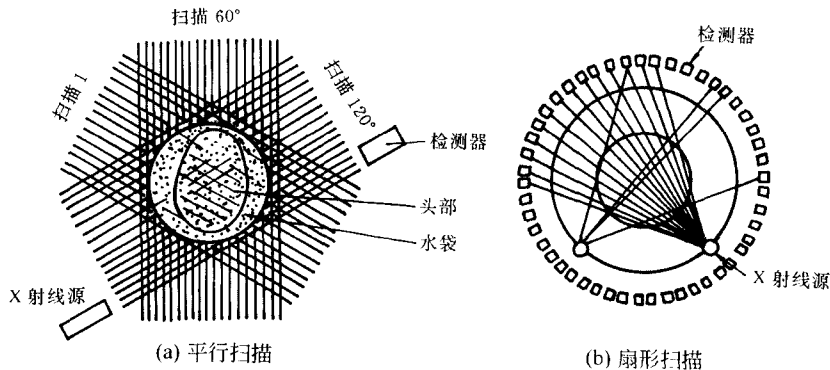


图1 扫描方式

根据投影数据重建被测断面图像的方法通常有五种:联立方程法,反投影法,傅里叶变换法,卷

积法,以及逐次逼近迭代法。联立方程法是早期的方法,也是最基本的方法,但计算比较繁琐。反投

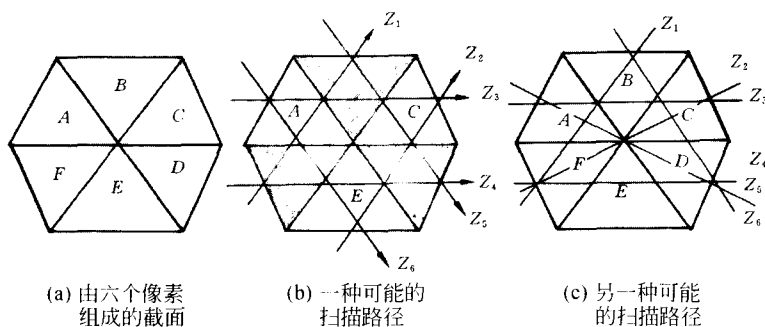


图2 联立方程法重建图像示意图

影法(或总和法)比较简单,但重建的图像质量较差。傅里叶变换法是利用对象投影数据的傅里叶变换与对象图像的傅里叶变换之间的关系来进行重建的方法,这种方法已被广泛地应用于断层摄影技术中。卷积法是由傅里叶变换法演变而来,这种方法的关键在于设计一个具有良好特性的重建滤波器。逐次逼近迭代法的速度比解方程组法要高,比反投影法有较高的精度,所以也被广泛采用。下面,我们介绍联立方程法及傅里叶变换法的基本概念。

(1) 联立方程法 为了说明概念,假定被测对象的截面只有六个像素(参见图像表示),如图2(a)所示。采用了一种扫描方法获得的数据表示在图2(b)中。设这六个像素的未知密度为 A, B, \dots, F , 而测得数据为 Z_1, Z_2, \dots, Z_6 。注意,这里的未知密度代表了对象物质对扫描射线的吸收程度,在图像中反映为灰度。由图2(b)可获得六个方程

$$\begin{aligned} A + B + F &= Z_1 & C + D + E &= Z_2 & A + B + C &= Z_3 \\ D + E + F &= Z_4 & B + C + D &= Z_5 & A + E + F &= Z_6 \end{aligned}$$

我们希望从这六个方程解出未知的 A, B, \dots, F , 但是这些方程并不独立,所以不能获得结果。改进的方法可以是增加扫描次数或改变扫描路径。例如,当采用图2(c)中的扫描方法时,未知的 A, B, \dots, F 就可以被解出来,因为有6个独立的方程:

$$\begin{aligned} A + B + F &= Z_1 & C + F &= Z_2 & A + B + C &= Z_3 \\ D + E + F &= Z_4 & A + D &= Z_5 & B + C + D &= Z_6 \end{aligned}$$

(2) 傅里叶变换法 根据投影图像的傅里叶变换与原图像傅里叶变换之间的关系(此处不作证明),可利用下列步骤从投影数据重建图像。

① 确定某方向 θ , 如图3中所示。从 t 方向获得投影数据称之为 $g(s, \theta)$ 。这里的 s 就是沿着图中所示的 s 方向的坐标。

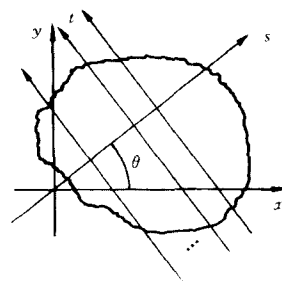


图3 傅里叶变换法重建图像示意图

② 在极坐标下求 $g(s, \theta)$ 的傅里叶变换, 即

$$F(R, \theta) = \int g(s, \theta) e^{-j2\pi R s} ds$$

这里, $s = x \cos \theta + y \sin \theta$, R 为傅里叶变换的频率。

③ 利用下式进行反变换

$$f'(x, y; \theta) = \int |R| F(R, \theta) e^{j2\pi R(x \cos \theta + y \sin \theta)} dR$$

④ 对所有不同 θ 方向的 $f'(x, y; \theta)$ 求和, 就获得重建的图像 $f(x, y)$

$$f(x, y) = \int_0^\pi f'(x, y; \theta) d\theta$$

这里介绍的是数据连续的公式。在实际的计算机处理过程中, 需要进行离散采样并进行离散的傅里叶变换。

参考文献

Herman G T. Image Reconstruction From Projections. Academic Press, 1980 (濮群)

tuxiang chuli de jiben yunsuan

图像处理的基本运算 (basic operations in image processing) 指图像处理中常用的点运算、邻域运算、几何运算、变换运算以及形态学运算等。这些基本的运算是图像处理和图像分析中经常用到的对图像作预处理的方法。图像携带了视觉(或其

他)信息,但是由于条件限制、环境干扰,或不同应用的特殊要求,图像“质量”往往不理想。为了有助于人或机器理解图像中的信息,常常有必要对图像作某种预处理。这些预处理包括改变图像的灰度分布,有意识地增强某些特征,或对噪声进行过滤,以使图像更适合于人眼的观察、理解,或便于机器的分析。图像预处理的结果仍为图像。在图像处理分析的系统中,这些预处理被称为低层图像处理(参见**图像并行处理**)。从预处理方法的角度来看,有以下几种基本运算。

(1) 点运算 这种运算的处理结果 $g(x, y)$ 只和本像素点 (x, y) 的值 $f(x, y)$ 有关(参见**图像点运算**),即

$$g(x, y) = T_p[f(x, y)]$$

其中 T_p 表示灰度变换运算。或

$$g(x, y) = T_{mp}[f_1(x, y), f_2(x, y), \dots]$$

其中 T_{mp} 为基本的算术或逻辑操作(+, -, ×, /, AND, OR, XOR 等)。

(2) 邻域运算 是空间域滤波方法。像素点 (x, y) 的处理结果与 (x, y) 的邻域 S 内的像素点的值有关(参见**图像邻域运算**),即

$$g(x, y) = T_n[f(m, n)], \quad (m, n) \in S,$$

S 为点 (x, y) 的邻域。

最常用的邻域运算是用卷积运算实现的线性滤波:

$$g(x, y) = f(x, y) * h(x, y)$$

(3) 变换运算 将空间域图像 $f(x, y)$ 通过变换,例如傅里叶变换,变换为频域函数 $F(u, v)$,再进行滤波或处理。在处理后,通常可以再施行反变换回到空间域(参见**图像变换运算**)。

(4) 几何运算 几何运算实质上是图像的坐标变换(参见**图像几何运算**)。经过几何运算,像素 $f(x, y)$ 的坐标变为 (x', y') 。几何运算的一般表达式为

$$x' = g_1(x, y), \quad y' = g_2(x, y)$$

(5) 彩色运算 包括伪彩色增强和彩色变换。伪彩色增强将灰度图像 $f(x, y)$ 映射为彩色图像 $[R(x, y), G(x, y), B(x, y)]$, $R(x, y) = F_r[f(x, y)]$, $G(x, y) = F_g[f(x, y)]$, $B(x, y) = F_b[f(x, y)]$, 其中 F_r, F_g, F_b 为映射函数。

(6) 形态学运算 利用数学形态学的基本运算对图像进行处理,从而达到改善图像质量的目的。这是不同于空域滤波和频域滤波的一种新的图像滤波方法,称为形态域滤波方法(参见**图像形态学运**

算)。一般的表达式为:

$$X' = M(X, B)$$

其中, X 为处理对象, B 为结构元素, 而处理结果是 X' ; M 表示形态学运算。

以上列举的基本运算不仅使用在机器人视觉、医学图像处理、遥感图像处理、文字识别等图像分析领域,而且广泛应用于影视特技、多媒体会议、虚拟现实等计算机应用领域中。在较为完整的图像处理系统中,一般都具有实现上述图像处理基本运算的硬件。

参考文献

1. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989
2. Castleman K R. Digital Image Processing. Prentice Hall, 1996 (朱志刚)

tuxiang de yasuo bianma

图像的压缩编码 (compression and coding of images)

为满足图像在存储、传输和应用方面的要求,对静止或运动的数字图像进行数据压缩和编码的过程、方法和技术。

图像有静止图像和运动图像之分。前者指的是单幅图像,后者指的是内容随时间变化的一个图像序列,也就是日常所说的视频,通常每秒钟包含 25~30 幅图像。

单幅图像的数据量可按下面的公式计算(以比特为单位):

图像数据量 = 图像水平分辨率 × 图像垂直分辨率 × 每个像素的比特数

可见其数据量很大。为了节省存储数字图像时所需要的存储器容量,降低存储成本,特别是在因特网应用中,为了提高图像的传输速度,减少通信费用,大幅度压缩图像的数据量是非常重要的。以使用电话接入因特网的用户为例,假设数据传输速率为 56 kb/s,则理想情况下,传输一幅分辨率为 640 × 480 的 6.5 万色的未经压缩的图像需要 1~2 min,如果图像的数据量压缩 10 倍,那么下载时间仅需 10 s 左右。

由于数字图像中的数据相关性很强,或者说,数据的冗余度很大,因此对数字图像进行大幅度的数据压缩是完全可能的。而且,人眼的视觉有一定的局限性,即使压缩前后的图像有一定失真,只要限制在人眼无法察觉的误差范围之内,也是允许的。

图像数据压缩可分成两种类型,一种是无损压缩,另一种是有损压缩。无损压缩是指压缩以后的数据进行图像还原(也称为解压缩)时,重建的图像

与原始图像完全相同。例如行程长度编码(RLC)、哈夫曼编码等。有损压缩是指使用压缩后的数据进行图像重建时,重建的图像与原始图像虽有一定的误差,但不影响人们对图像含义的正确理解。

图像的压缩方法很多,不同方法适用于不同的应用,在计算机中常常是多种压缩方法的综合使用。为了得到较高的数据压缩比,数字图像的压缩一般都采用有损压缩,如变换编码、向量编码等。评价一种压缩编码方法的优劣主要看3个方面:压缩倍数的大小,重建图像的质量(有损压缩时),以及压缩算法的复杂程度。

为了便于在不同的系统中交换图像数据,人们对计算机中使用的图像压缩编码方法制定了一些国际标准和工业标准。国际标准化组织(ISO)和国际电工委员会(IEC)两个国际机构联合组成了专家组,负责制定了一组静止图像数据压缩编码的国际标准。如连续色调静止图像的编码标准 JPEG(ISO/IEC 10918)和 JPEG 2000(ISO/IEC 15444),二值(黑白)图像的编码标准 JBIG1(ISO/IEC 11544)和 JBIG2(ISO/IEC 14492)等。其中 JPEG 的适用范围较广,它能处理各种连续色调的彩色或灰度图像,算法复杂度适中,既可用硬件实现,也可用软件实现。JPEG 图像的压缩比是用户可以控制的。压缩比越低,图像质量越好;压缩比越高,图像质量越差。对于内容为自然风光的彩色图像,压缩比为 10~20 时重建图像的误差很难察觉。

最新的静止图像压缩编码的国际标准是 JPEG 2000,它是为了满足 21 世纪图像应用的需要而开发的,适用于各种不同类型(黑白、灰度、彩色等)和不同特性的图像(自然图像、医学图像、遥感图像、合成图像等),可应用于各种不同的应用模式(实时传输模式、检索应用模式、存档应用模式等)。由于采用了小波分析等先进算法,因而提供了许多 JPEG 所不具备的功能,如更好的图像质量,更低的码率,更适合在万维网上传输等,它与 JPEG 保持向下兼容。

运动图像(视频)的数据量比静止图像大得多。例如 1min 的 CCIR 601 规格(演播室质量)的数字视频,其数据量约为 1 GB,这样大的数据量无论是存储、传输还是处理,都是极大的负担,为此必须对数字视频进行大幅度的数据压缩。

由于视频信息中相邻画面的内容有高度的连贯性,其变化往往发生在画面的局部范围。因此,可以对相邻画面中发生变化的部分用运动向量来描述,这样,后续的图像就可以看成是它前面图像经过运动补偿后的结果。基于这种运动补偿的画面预测和画面插值的方法是数字视频压缩编码的主要手段。再加上各个画面内部有很强的信息相关性,所以数字视频的数据量可压缩几十倍至上百倍。视频信息压缩编码的方法很多,一个好的方案往往是多种算法的综合运用。目前,国际标准化组织制定的有关数字视频(及其伴音)压缩编码的几种标准及其主要应用可参见表 1。

表 1 视频压缩编码的标准及其应用

名 称	原图像大小	压缩后的码率	主要应用
MPEG-1	360×288	1.2~1.5 Mb/s	适用于 VCD、数码相机、数码摄像机等
H. 261	360×288(CIF), 180×144(QCIF)	P×64 kb/s (P=1,2 时,只支持 QCIF 格式;P≥6 时,可支持 CIF 格式)	应用于视频通信,如可视电话、会议电视等
MPEG-2 (MP@ML)	720×576	5~15 Mb/s	用途最广,如 DVD、150 路卫星电视直播、540 路 CATV 等
MPEG-2 High Profile	1440×1152 1920×1152	80~100 Mb/s	高清晰度电视(HDTV)领域
MPEG-4 (H. 324)	多种不同的视频 格式	与 MPEG-1, MPEG-2 相当,但最低可达到 64 kb/s	适合于交互式多媒体应用,包括虚拟现实、远程教学、交互式电视等

参考文献

1. 国际标准化组织 JPEG 网站: <http://www.jpeg.org/>

[jpeg.org/](http://www.jpeg.org/)

2. 国际标准化组织 MPEG 网站: <http://www.mpeg.org/>

telecomitalialab.com/

(张福炎)

tuxiang dian yunsuan

图像点运算 (image point operation) 将图像各点的灰度逐点变换成另一灰度的运算。根据一幅图像进行的灰度变换,一般采用两种技术:一是根据事先定义的灰度映射关系进行变换的技术,二是根据图像直方图统计所进行的直方图修正技术。对于两幅或两幅以上图像对应点的点运算称为图像的运算(包括算术和逻辑运算)。

灰度变换 灰度变换是通过函数关系 $g = T(f)$ 把给定的灰度级 $f \in [0, L]$ 映射到另一灰度级 $g \in [0, L']$ 的方法,常用的灰度变换有:

(1) 对比度扩展 当光照不足或不均匀,或者成像系统的动态范围小时,都会产生低对比度的图像。所谓“低对比度”的图像就是图像灰度集中在一个小范围内。当知道在某个灰度范围内的像素数目较多时,扩展该灰度范围可增强整个图像的视觉效果。

下面是对比度扩展灰度变换的一个例子,其变换为

$$g = \begin{cases} \alpha f, & 0 \leq f < a \\ \beta(f - a) + ga, & a \leq f < b \\ \gamma(f - b) + gb, & b \leq f \leq L \end{cases} \quad (1)$$

其中, α, β, γ 是直线斜率。如果需要扩展原图像的灰度区域 $[a, b]$, 则斜率 β 应该大于 1。

(2) 阈值化 阈值化常用于文字识别中的文字分割预处理。由于光照或传感器方面的噪声,输入的白纸黑字图像并不是黑白分明的。利用阈值化运算可得到黑白分明的二值化的图像。例如,下面是利用阈值 t 进行二值化的灰度变换函数,即

$$g = \begin{cases} 0, & f < t \\ L, & f \geq t \end{cases} \quad (2)$$

(3) 灰度反转 通过灰度反转可以得到输入图像的负像。灰度反转可用于医学图像处理、显示以及图像的打印等。灰度反转的变换函数可以是

$$g = L - f, \quad g, f \in [0, L] \quad (3)$$

(4) 灰度窗口切分变换 将具有某一区间的灰度级的像素与其他部分(“背景”)分开,其表达式可以是

$$g = \begin{cases} L, & a \leq f \leq b \\ 0, & \text{其他} \end{cases} \quad \text{清除背景} \quad (4)$$

$$g = \begin{cases} L, & a \leq f \leq b \\ f, & \text{其他} \end{cases} \quad \text{保留背景}$$

其中 $[a, b]$ 为所关心的灰度窗口。利用这种变换可检测出具有某一灰度区间内灰度级的所有像素,是图像灰度分析的常用方法。

(5) 灰度范围压缩 有时图像的灰度范围过大,以至于只有在某一灰度范围内的少量的像素能够被分辨出来。灰度范围一般可以通过对数变换来进行压缩,例如,

$$g = c \lg(1 + f) \quad (5)$$

其中 c 为一比例系数。对数变换压缩了灰度值高的范围,对灰度值低的范围反而有所增强。

(6) 查找表(LUT) 查找表常用于图像获取和显示的硬件设备中(参见图像获取)。由于灰度变换与像素位置无关,故可以通过查找表(LUT)快速实现。实现方法可将原灰度值作为 LUT 的地址, LUT 则根据某一灰度变换函数 $g = T(f)$ 来构成。LUT 的输出为变换后的灰度 g 。因为只需计算和装载 LUT 一次,利用 LUT 可以避免计算每个像素点的灰度变化,便于快速处理。

直方图修正 灰度直方图表示了每种灰度在图像中出现的频率。灰度直方图修正技术是通过将直方图修改为所希望的形状以达到改善图像的目的。

(1) 直方图均衡化 直方图均衡化就是把给定图像的直方图分布改变成均匀直方图分布。直观地讲,直方图均衡化导致了图像的对比度增加。需要注意的是,由于灰度等级的离散化,均衡化图像的直方图只是近似均匀分布。均衡化后图像的灰度范围扩大了,但其本质是扩大了一些量化层之间的间隔,而不是量化层的数目。有时,对整幅图像的直方图均衡化可能会抑制一些像素点不多但有用的细节。为解决这个问题均衡化可逐块进行,所分的块可以交叠或不交叠。分块的方法尤其适用于不均匀的图像。

(2) 直方图规定化 均衡化的方法是直方图修正技术的一个特例,即希望修正后的图像灰度直方图为均匀分布。但在某些应用中,我们可能希望达到预先给定的分布,以便有意地强调某些灰度级的范围。这种方法称为直方图规定化。

图像代数运算 图像代数运算是两幅以上图像的运算。在一些图像系统中常用硬件来实现。图像代数运算可用于图像平滑、运动检测、逐点校正和逐点逻辑判断等。

(1) 多图像平均与图像平滑 这种方法用于同一目标的图像可以重复拍摄の場合,目的是抑制图

像摄取过程中由于种种原因引入的噪声。

(2) 图像相减与变化检测 在许多应用中,经常需要快速比较两个内容复杂的图像。一个简单的办法是将两幅对准的图像相减,这样它们之间的差别将会被强调出来。它可应用于医学诊断检测,安全监视,路面障碍物检测等。

(3) 图像相乘(除)与灰度校正 当图像灰度与实际景物亮度不匹配(或不一致)时,如果知道造成这种不一致是由于光照的不均匀或由于摄像机各点灵敏度的差异造成的,那么可以用逐点灰度校正,即把原图乘(或除)以一个校正系数图像的方法来处理。

参考文献

1. 余松煜,周源华,吴时光. 数字图像处理. 北京:电子工业出版社,1989
2. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989 (朱志刚)

tuxiang fanxiang lübo fuyuan

图像反向滤波复原(image restoration by inverse filtering) 针对图像退化中的散焦退化现象的一种图像复原的过程和方法。参见图像复原条目中式(4)

$$G(u,v) = \frac{F(u,v) - N(u,v)}{H(u,v)}, \quad u,v = 0,1,\dots,m-1$$

式中, $F(u,v)$, $G(u,v)$, $N(u,v)$ 分别为退化图像 $f(x,y)$, 原始图像 $g(x,y)$ 及噪声 $n(x,y)$ 的傅里叶变换。 $H(u,v)$ 则是系统点扩散函数 $h(x,y)$ 的傅里叶变换。

当噪声 $N(u,v)$ 小到可以忽略时,即 $N(u,v) \approx 0$ 时,

$$G(u,v) = \frac{F(u,v)}{H(u,v)}, \quad u,v = 0,1,\dots,m-1 \quad (1)$$

对 $G(u,v)$ 进行傅里叶逆变换即可得到原始图像,即

$$g(x,y) = F^{-1} \left[\frac{F(u,v)}{H(u,v)} \right], \quad x,y = 0,1,\dots,m-1$$

这种复原方法称为反向滤波法复原。这一名词的来由是把 $H(u,v)$ 当作一个“滤波”函数,它乘以 $G(u,v)$ 得 $F(u,v)$ 是退化图像的傅里叶变换; $F(u,v)$ 除以 $H(u,v)$ 形成反滤波得到 $G(u,v)$, 对 $G(u,v)$ 再作一次傅里叶逆变换,便复原出原始图像 $g(x,y)$ 。

上面分析是在 $N(u,v) = 0$ 的条件下,实际上 $N(u,v)$ 经常不为零。有噪声时,反向滤波得出的是

原始图像的近似值 $\hat{g}(x,y)$ 。在频率域中,原始图像的近似值的傅里叶函数为

$$\hat{G}(u,v) = G(u,v) + \frac{N(u,v)}{H(u,v)}, \quad u,v = 0,1,\dots,m-1 \quad (2)$$

注意,近似值 $\hat{G}(u,v)$ 与准确值 $G(u,v)$ 的差是 $N(u,v)/H(u,v)$ 。 $\hat{G}(u,v)$ 经傅里叶逆变换得到 $\hat{g}(x,y)$ 。

图1给出一个用反向滤波法与一个含有噪声模糊图像复原的例子。

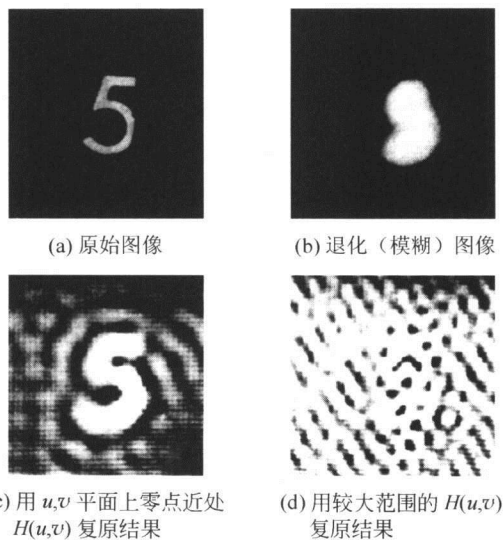


图1 用反向滤波进行图像复原

(此图取自参考文献3)

将图1中的(c)和(d)比较,显然(c)的效果比(d)好。这是由噪声项 $N(u,v)/H(u,v)$ 引起的。考虑到 $H(u,v)$ 随着 u,v 频率变量增加而衰减,但 $N(u,v)$ 变化不大的特点,(c)只计算零点附近低频区的 $G(u,v)$, 因为 $G(u,v)$ 和 $H(u,v)$ 在低频区的值都较大,所以 $N(u,v)/H(u,v) \ll G(u,v)$, 复原效果较好。(d)在较大范围内取 $G(u,v)$, 包括了 $H(u,v)$ 趋于零的值,而 $G(u,v)$ 已衰减,所以 $N(u,v)/H(u,v) \gg G(u,v)$ 。由于噪声淹没了图像,导致了复原失败。由此可见,用反向滤波进行图像复原不适合于信噪比小的图像。

参考文献

1. 赵荣椿等. 数字图像处理导论(第二版). 西安:西北工业大学出版社,1995
2. Jain A K. Fundamentals of Digital Image Processing. Prentice-Hall Inc., 1989

3. Rafael C, Gonzalez/Paul Wintz. Digital Image Processing (Second edition). Addison-wesley, 1987

(李树青)

tuxiang fenge

图像分割 (image segmentation) 将一幅图像分成不同的部分或区域的过程和方法。图像分割是在**数字图像处理**、图像分析及图像理解中的一个重要的基本步骤。其目的是把图像分成一些有用的或有意义的部分或区域,以便进一步对图像进行分析与理解。所谓有意义的区域是指与应用有关的区域。例如,在一张卫星拍摄的地球的图像(照片)上,把水域与陆地分割开来;在一张田野的照片上,把农田与道路分割开来。可以说,图像分割是对图像进行分析理解的关键步骤。例如,对图像中的文字进行识别,需要首先分割出字符,然后才能识别。又例如,利用图像对工厂中的工件进行识别,也需要首先把工件与背景分开。

图像分割的基本理论与方法是和**模式识别**紧密相关的。也可以说,图像分割是模式识别在图像处理及图像理解中的应用。图像分割通常需要一些所谓的先验知识或假定。例如,如果事先知道在一张田野的照片里,只有农田和道路,就可以将这幅图像分成两类区域。但如果还有一些建筑物在这幅图像里,可能就要考虑多于两类的分法。进一步说,如果知道农田是较暗的绿色而道路是较明亮的黄色时,那么分割的任务就会容易一些,图像中分割的区域就会准确一些。

从图像处理的角度来看,图像分割通常可根据两种原则来进行。一是根据各图像像素之间的不同灰度或不同的颜色分量来进行分割。这称之为基于像素的分割方法。另一种是基于不同类型的区域在图像中的不连续性来进行分割,称之为基于区域的分割方法。例如,可以根据不同区域有不同的纹理(组织结构特征)来进行分割。此外,图像中的不连续也造成了边界,也可以利用边界来进行图像分割。基于像素的方法可以用来分割白纸黑字,或黄色的墙和棕色的门。但是,要分割一幅图像中的两种不同的花布,就要求助于基于区域(纹理组织)的方法了。基于像素的分割方法还可以再分为利用门限或直方图的方法(参见**图像像素分类**)。基于区域的分割方法还可以再分为利用边界(边缘)、利用区域以及同时利用边界和区域的方法(参见**图像特征提取**、**图像边缘检测**、**图像轮廓表示**及**图像区域**

分割)。

参考文献

1. 余松煜,周源华,吴时光. 数字图像处理. 北京: 电子工业出版社, 1989

2. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989 (俞志和)

tuxiang fenxi

图像分析 (image analysis) 用计算机对图像中一些感兴趣的对象特性(如形状、纹理、结构等)作分析,并提取其描述信息的过程。

图像分析与图像处理的研究内容密切相关,并无严格的分界线。多数人认为后者的研究内容偏重于**图像复原**、**图像增强**、**图像压缩**等,处理结果的表现形式仍是二维图像(即图像→图像)。而图像分析则是在处理的基础上作进一步分析,其结果是有关图像的描述或者感兴趣对象的特征(图像→描述)。有些人把上述过程都归入图像处理领域,把图像分析作为图像处理的一个阶段。具体来说,图像分析的主要内容大致包含**图像边缘检测**、**图像区域分割**、**图像特征提取**及**图像序列分析**(参见**运动分析**)等方面。要进一步了解可参见上述条目。

参考文献

1. 徐建华. 图像处理与分析. 北京: 科学出版社, 1992

2. 吴建康. 数字图像分析. 北京: 人民邮电出版社, 1989

3. Rosenfield A, Kak A C. Digital Image Processing. 2nd. ed. U. S. A. : Academic Press, 1982

(阎平凡)

tuxiang fuyuan

图像复原 (image restoration) 对一些退化或劣化的图像进行校正处理、滤除退化痕迹、恢复图像本来面目的过程和技术。又称图像恢复。图像复原应尽可能复现或逼近无退化真实图像。

图像复原技术和**图像增强**技术都是用以改善图像质量的图像处理技术,但是两者之间有以下差别:图像增强常以人机交互方式用某种试探性方法提高图像的质量,以改善人眼的视觉效果和主观感觉,或突出图像中的某些特征以便计算机更好地识别和理解图像;而图像复原强调的是尽可能客观地以最大保真度恢复图像本来面目。图像复原的质量

不是由人的主观心理感觉决定的,而是根据某些客观的质量标准来决定的。

图像复原可以认为是**图像退化**的逆过程。对于一幅 $m \times m$ 点的数字图像 $g(x,y)$ (原始图像) 经成像光学系统作用后得退化图像 $f(x,y)$, 数学上可以用 $g(x,y)$ 与光学系统点扩展函数 $h(x,y)$ 的卷积计算, 即

$$\begin{aligned} f(x,y) &= g(x,y) * h(x,y), \\ x,y &= 0,1,\dots,m-1 \end{aligned} \quad (1)$$

再加上噪声 $n(x,y)$ 得

$$\begin{aligned} f(x,y) &= g(x,y) * h(x,y) + n(x,y), \\ x,y &= 0,1,\dots,m-1 \end{aligned} \quad (2)$$

而由模糊图像复原到原始图像直接由式(2)求解, 即复原 $g(x,y)$ 是很困难的。但利用傅里叶变换(参见**图像变换**)并根据卷积定理可得出

$$\begin{aligned} F(u,v) &= G(u,v)H(u,v) + N(u,v), \\ u,v &= 0,1,\dots,m-1 \end{aligned} \quad (3)$$

从而

$$\begin{aligned} G(u,v) &= \frac{F(u,v) - N(u,v)}{H(u,v)}, \\ u,v &= 0,1,\dots,m-1 \end{aligned} \quad (4)$$

其中, $F(u,v)$, $G(u,v)$, $N(u,v)$ 分别为退化图像 $f(x,y)$ 、原始图像 $g(x,y)$ 及噪声 $n(x,y)$ 的傅里叶变换。 $H(u,v)$ 则是系统点扩展函数 $h(x,y)$ 的傅里叶变换, 称为成像系统的传递函数。这里的 $H(u,v)$ 与图像退化中的算子 H 形式相似, 但含义不同。

对 $G(u,v)$ 进行傅里叶逆变换, 即可得到所需要的原始图像。

图像复原有 3 种方法: **图像反向滤波复原**、**图像维纳滤波复原**和**图像运动模糊复原**。

参考文献

1. 赵荣椿等. 数字图像处理导论(第二版). 西安: 西北工业大学出版社, 1995
2. Jain A K. Fundamentals of Digital Image Processing. Prentice-Hall Inc., 1989
3. Rafael C, Gonzalez/Paul Wintz. Digital Image Processing (second edition). Addison-Wesley, 1987

(李树青)

tuxiang gujia biaooshi

图像骨架表示 (image skeleton representation) 利用图像骨架这种结构关系来描述图像中区域的方法。区域的中心骨架可以通过图像细化及中轴变换等方法来获取。

所谓中轴就是区域的对称轴线, 中轴变换就是寻找对称轴线的过程。如果把轮廓线上的点称为轮廓点, 那么骨架就是离两个或两个以上最近轮廓点距离最大的点之集合。

设一个区域为 R , 其轮廓线为 B 。对于 R 中的每个点 p , 可找到其在 B 上的最近相邻点。如果 p 有多个这种相邻点, 这些点即构成了 B 的中轴或骨架。显而易见, 中轴的计算是与距离的定义直接相关的, 常用的距离定义有欧几里得距离等。

为了确定骨架点, 可以想象所考虑的区域是一片干草地, 而区域边界的外面是池塘, 当在区域的边界同时点上火以后, 火就会烧进区域里面去。假如火的蔓延速度一样, 从边界烧进来的火相遇的点就是骨架点。图 1 是 3 种不同的形状与它们的骨架。

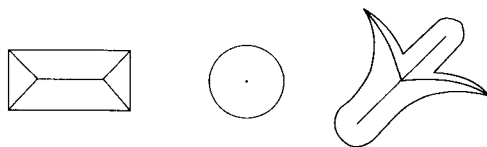


图 1 骨架

利用形态学运算也可以获得区域的结构特征及骨架(参见**图像形态学运算**)。

参考文献

1. 赵荣椿等. 数字图像处理导论(第二版). 西安: 西北工业大学出版社, 1995
2. Gonzales R, Woods R. Digital Image Processing. Addison-Wesley Publishing Company Inc., 1992

(曾建超)

tuxiang huoqu

图像获取 (image acquisition) 利用图像输入设备(例如摄像机)将图像输入计算机的过程和技术。常用的图像输入设备有摄像机、线阵摄像机、扫描仪等。传统的输入设备可以产生景物或图像的模拟信号, 模拟信号经数字化后输入计算机(参见**图像表示**)。随着技术的进步, 许多设备可以直接产生数字图像, 这些设备中的图像传感器往往具有数字化功能。

模拟输入 尽管不同的图像采集设备的精度、速度及成本可能大不相同, 但其原理却是相同的。在一个图像采集设备中, 光学系统将景物或图片的“像”聚焦到成像面上。传感器将投射到其上面的

光信号转换为电信号。传感器的输出部分将该电信号缓冲和放大,必要的话,还要转换为适于采集的电压(如全电视信号为1V)。该电压经过**模数转换器**变成数字信号。时序和地址逻辑保证了输入模数转换器信号按适当的周期(时间间隔)采集,并存入图像存储器的正确位置。如果数字化的是彩色信号,则对于图像每个像素,传感器返回三个值,即红、绿、蓝分量。因此图像获取设备的基本部件为光学系统、传感器、模数转换器和图像存储器(帧存储器)。

电视摄像机 电视摄像机一般都装有一次能够扫描一整幅“图片”的电子线路。这种工作方式与广播标准和电视接收机标准密切相关,它更面向于人的观察方式而不是计算机图像处理所采用的方式。一整幅图像(我国采用的PAL制式中是625行扫描线,美国NTSC制式中是525行)为一帧,一帧包含两场,即分别由一帧中奇数行扫描线和偶数行扫描线组成的奇数场和偶数场,这两场信号由摄像机电子线路顺序产生并传递。因此被传送的图像是交替的,奇数场先“画”在屏幕上,然后是偶数场。隔行扫描是为了防止图像闪烁。

扫描摄像机 扫描摄像机一次只获取一行图像。为了获取整幅图像,传感器必须用机械方法扫过所需视野,时间从几秒钟到数分钟,依所要求的信噪比而定。扫描摄像机不受电视标准的限制,能提供很宽的采集速度选择范围,并可获得高质量的图像。

线阵摄像机 和扫描摄像机非常类似,但传感器和光学系统是固定在一起的,如果需要扫描,则由摄像机和物体的相对运动形成,一般情况下是物体移过摄像机的视野。线阵摄像机成像和产生数据的速度很高,常用于工业生产中。

文件扫描仪 是扫描摄像机的另一应用。它可以获取静止的、平面的、大幅的高质量图像,用于文字和图纸输入。

数字输入 随着数字图像获取技术的发展,直接获取数字形式数据的方法越来越普遍。其中有3个领域应用最广,即遥感成像、医学图像和距离成像。近来,在多媒体计算技术中也越来越普遍采用直接输出数字信号的摄像机。这些直接产生数字图像的数字照相机或摄像机有CCD和CMOS两种传感器,有些摄像机的成本很低,由于可以通过USB或其他标准接口直接输入计算机,已被广泛使用。

遥感成像 包括航空和卫星成像。常用的卫星

图像使用产生数字信号的传感器。

医学成像 用数字形式记录医学图像始于用X光构造人体内组织的断层图像(参见**图像重建**)。这种技术称为计算机断层摄影,简称CT。另一种是核磁共振成像(MRI),它利用原子核在强磁场下对电磁波的吸收产生图像。不论哪种方法都不直接测量图像数据,而是由传感器的输出计算出图像。

距离成像 距离成像的主要方法是雷达成像,其中包括超声雷达,激光雷达等。这种成像系统获取的数据是传感器到目标的距离而不是物体的亮度。距离是通过计算得到的,因此它产生数字形式的图像。

近年来全方位成像在视觉导航、视觉监视和视频消费领域得到不少应用,其特点是一次实时获取360°全方位的图像。使用鱼眼透镜、锥面反射镜或特殊设计的全景镜头都可以拍摄到360°全方位图像。

参考文献

1. Law A. Introductory Computer Vision and Image Processing. McGraw Hill Book Company, 1991
2. Yagi Y, Kawato S, Tsuji S. Real-time omnidirectional image sensor (COPIS) for vision-guided navigation. IEEE Trans Robotics and Automation, 1994, 1(1):11~27
3. 朱志刚. 全视野时空视觉导航——真实景物的成像、建模与表示. 北京: 高等教育出版社, 2001 (朱志刚)

tuxiang jihe tezheng biaooshi

图像几何特征表示 (image geometric feature representation) 将图像中的物体或物体边界的几何特征用合适的数据结构或数学式表示的方法。这里的几何特征包括面积、紧凑性、偏心率、欧拉数、斜率密度函数等。

面积 设区域轮廓由 $(x(s), y(s))$ 表示,其中 s 为弧长参数,则区域面积由格林公式计算。

$$\text{面积} = \int_0^l \left(x \frac{dy}{ds} - y \frac{dx}{ds} \right) ds \quad (1)$$

其中 l 是区域的周长。

紧凑性 紧凑性 = (周长)²/面积

这是一个无量纲的量。显然,圆形区域的紧凑性为最小值,表明其紧凑性最好。

偏心率 定义为区域的最长弦 A 与其垂直方向上的最长弦 B 之比(参见图1),即

$$\text{偏心率} = A/B$$

(2)

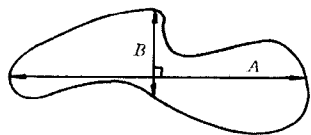


图1 偏心率

欧拉数 这是一个表示区域连通性的几何特征。它定义为

$$(\text{连通域的个数}) - (\text{孔的个数})$$

斜率密度函数 虽然被称为斜率密度函数,实际上这是利用切线角密度来表示轮廓的方法。切线角密度可以用直方图来表示,横坐标表示轮廓切线的角度,纵坐标表示具有某个切线角的切线数。显然,圆的斜率密度函数为一条水平直线,而折线轮廓斜率则会出现间断。利用斜率密度表示的例子见图2。

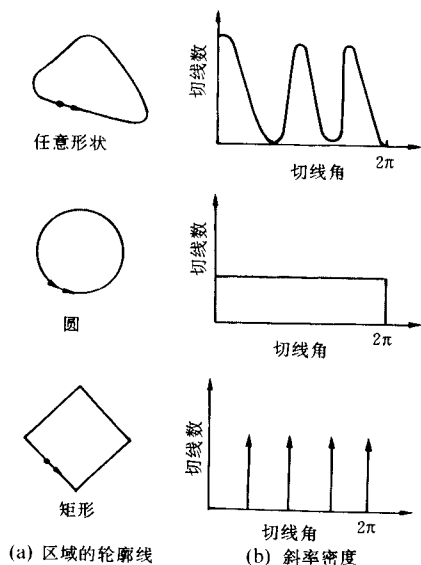


图2 斜率密度函数表示

除了上述的几何特征外,还有对称性,形状数等一些有用的几何特征。

参考文献

Rosenfeld A, Kak A C. Digital Picture Processing. Academic Press, 1982

(曾建超)

tuxiang jihe yunsuan

图像几何运算 (image geometric operation)

改变图像大小、形状、位置等几何属性的运算。图像几何运算包括图像重排序、图像缩放、剪贴、平移旋转和变形等。

图像重排序 通过改变图像像素的排序不仅可以产生一些有意义的平面或空间的显示效果,而且有时还有助于数据的处理或理解。例如,镜像变换可用于对称图像的比较,比较由光学成像引起的镜像(如胶片或投影片反置),以及医学中对称部位的图像(如左右肺)等。**图像转置**就是把图像的行、列交换,它在涉及向量及矩阵的几何运算中有用。例如,转置可用在图像的快速傅里叶变换中对行和列分别进行计算(参见图像变换)。**图像旋转** $K \times 90^\circ$ ($K=1,2,3$)可以通过简单的像素重排序得到,除了显示效果外, 90° 旋转可用于不同方向排列的文字识别以及模板卷积的方向配合等。

图像缩放 对于数字图像,改变图像的大小就意味着改变每行、每列像素的个数,即改变分辨率。改变图像大小的一个最明显的目的是为了在特定比例的显示器上或打印设备上呈现图像。另外,不同分辨率图像在图像分析中也有重要的应用。放大图像时需要生成新的像素,一般可以利用邻域的像素值来估计新的像素值,这个操作称为插值。当缩小图像时,需要丢弃一些像素。这时,既可以简单地丢弃原来的一些像素,也可以考虑利用被丢弃的像素值来修改保留的像素值。一般来讲,要尽量保持原图中的形状、亮度及对比度。所以,在改变像素的数目时,可能要在效果和计算复杂度上取一个折中。

坐标映射和重采样是改变图像尺寸的基本技术,其操作有两步:映射关系的确定和重新采样。当建立了原来像素坐标和结果像素坐标的映射关系后,可能产生具有带小数部分的坐标,这对于需要用整数坐标来表示的数字图像是个问题。因此,我们需要有第二个操作,即重采样。数字图像重采样和模拟数据采样有很大的不同。在数字图像重采样中,虽然我们不知道需要的采样点的位置,但是往往不能直接获得新坐标下的像素的亮度,因为可能没有与此位置对应的原图像的采样值。因此,我们必须根据已知的邻点值利用插值来获取需要的采样值。常用的插值方法有:

(1) 最近邻插值 结果图像中每个像素的值选择与其在原图中对应点最接近的像素的值。

(2) 线性插值 尽管线性插值只是一种近似,在图像处理中由于其效果很好,故使用很多。

(3) 曲线插值 当需改变尺寸的图像噪声较大

时,使用曲线插值更好。另一种方法是先进行图像平滑,再使用线性插值。

图像剪贴 在应用中经常需要取出图像的一部分单独进行处理,或者将两幅图像组合在一起等。这些操作分别称为**裁剪**和**拼贴**。裁剪操作是根据某种几何参数抽出一块图像,即子图像,而拼贴操作是将几幅图像或子图像合成一个新图像。剪贴不仅用于图像处理,在图像显现上也很有用。在一般情况下,裁剪的图像区域不一定是矩形,可以是圆形、三角形、星形和其他任意形状。为了定义裁剪区域,一般有三种方法:

(1) 参数形式 这对于规则的形状较适用,例如矩形可用其左上角和右下角表示,圆可用圆心和半径表示等。

(2) 区域边界表示 用一系列边界点来表示需要裁剪的区域,这在手工裁剪时常用。

(3) 标注方法 对于任意的裁剪区域,可以先用其最大外接矩形裁剪图像,然后对处于矩形区域内的像素加以标识,这样就形成一个标注图像,表示像素是否属于所关心区域。这种方法在图像生成和动画制作中很有用。另外,由于绝大部分图像系统只能储存矩形图像,所以对于非矩形图像需要通过一定的填补方式表示成一个矩形阵列。

图像平移和旋转 平移、旋转可以校正输入图像的位置和方向,以便放正图像或进行图像比较。为了实现图像的平移和旋转,可以应用有关平移和旋转的变换矩阵。在图像平移或旋转时还要考虑两个问题:①变换前后图像的范围;②变换后新的图像点的灰度取值问题。第一个问题可通过裁剪解决,第二个问题可以通过插值方法解决。

图像变形 虽然平移、旋转可用于校正两幅或多幅图像之间的位置或方向,但有些图像在成像过程或数字化过程中的畸变并不能用二维的平移或旋转来改善。所以,往往需用更通用的图像变形方法。二维图像透视变换可以校正因摄像机纯旋转所产生的任意三维景物的图像变形或摄像机任意运动所产生的平面物体的图像变形。图像变形可用于分析、理解图像或表现实验结果等方面。

图像卷曲 是一种常用的图像变形技术。在前述的几种修改图像的方法中,包括图像缩放、平面旋转和透视变换等,都是基于相同的基本操作,即像素坐标的映射和图像的重采样来进行的。不同方法的区别只是映射的方式不同。另外,在前述的几种基本方法中,映射在几何意义上是精确定义而且和图像

数据无关的。但是,更一般的复杂的图像映射有时不能够精确地定义,而且可能和图像本身有关,这种广义的几何变换称为图像卷曲,它可能既改变图像的大小又改变图像的形状。图像卷曲广泛地应用于遥感成像、视觉导航和虚拟现实等领域中。

参考文献

1. Lewis R. Practical Digital Image Processing. Ellis Horwood, 1990
2. Castleman K R. Digital Image Processing. Prentice-Hall, 1989 (朱志刚)

tuxiang ju biao shi

图像矩表示 (image moment representation)

把图像中的区域(物体的形状)用被称为“矩”的数学度量来表示的方法。由于图像区域的某些矩对于平移、旋转、尺度等几何变换具有一些不变的特性,所以矩的表示方法在物体(形状)分类、识别方面有重要意义。

设图像用 $f(x, y)$ 有界函数表示,其中 x, y 的定义域为 Ω ,则 $f(x, y)$ 的 $(p+q)$ 阶矩为

$$m_{p,q} = \int \int_{\Omega} f(x, y) x^p y^q dx dy$$

其中 $p, q = 0, 1, 2, 3, \dots$

当 $p = q = 0$ 时, $m_{0,0}$ 被称为图像 $f(x, y)$ 的零阶矩。它实际上是图像 $f(x, y)$ 灰度值的总和。对于二值图像来讲,如果灰度值是“1”表示物体,“0”表示背景,那么零阶矩 $m_{0,0}$ 就是图像中物体的面积。当 $p = 1, q = 0$ 时, $m_{1,0}$ 对二值图像来讲就是物体上所有点的 x 坐标的总和。类似地, $m_{0,1}$ 就是物体上所有点的 y 坐标的总和,所以

$$\bar{x} = \frac{m_{1,0}}{m_{0,0}}, \bar{y} = \frac{m_{0,1}}{m_{0,0}}$$

就是二值图像中物体质心的坐标。

当 $p = 2, q = 0$ 时, $m_{2,0}$ 就相当于物体绕 y 轴旋转的转矩。类似地, $m_{0,2}$ 相当于物体绕 x 轴的转矩。

为了获得矩的不变特征,往往采用中心矩以及归一化的中心矩。中心矩定义为

$$m'_{p,q} = \int \int (x - \bar{x})^p (y - \bar{y})^q f(x, y) dx dy$$

这里, \bar{x}, \bar{y} 是物体重心的坐标。

用中心矩表示的一阶矩恒为零。

$$m_{1,0} = m_{0,1} = 0$$

对于 $p+q = 2, 3, 4, \dots$ 的高阶矩,可以定义归一化的中心矩

$$u_{p,q} = \frac{m'_{p,q}}{(m'_{0,0})^r}, r = \left(\frac{p+q}{2} + 1 \right)$$

利用归一化的中心矩,可以获得一系列对于平移、旋转、尺度的变换的矩不变量包括:

$$\begin{aligned} \varphi_1 &= u_{2,0} + u_{0,2} \\ \varphi_2 &= (u_{2,0} - u_{0,2})^2 + 4u_{1,1}^2 \\ \varphi_3 &= (u_{3,0} - 3u_{1,2})^2 + (u_{0,3} - 3u_{2,1})^2 \\ \varphi_4 &= (u_{3,0} + u_{1,2})^2 + (u_{0,3} + u_{2,1})^2 \\ \varphi_5 &= (u_{3,0} - 3u_{1,2})(u_{0,3} + u_{2,1}) \\ &\quad \times [(u_{3,0} + u_{1,2})^2 - 3(u_{2,1} + u_{0,3})^2] \\ &\quad + (u_{0,3} - 3u_{2,1})(u_{0,3} + u_{2,1}) \\ &\quad \times [(u_{0,3} + u_{2,1})^2 - 3(u_{1,2} + u_{3,0})^2] \\ \varphi_6 &= (u_{2,0} - u_{0,2})[(u_{3,0} + u_{1,2})^2 \\ &\quad - (u_{2,1} + u_{0,3})^2] \\ &\quad + 4u_{1,1}(u_{3,0} + u_{1,2})(u_{0,3} + u_{2,1}) \end{aligned}$$

矩不变量已成功地应用于飞机形状的区分、字符识别以及景物匹配等。

参考文献

1. 余松煜,周源华,吴时光. 数字图像处理. 北京: 电子工业出版社, 1989
2. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989
3. Gonzales R, Woods R. Digital Image Processing, Addison-Wesley Publishing Company Inc., 1992
(曾建超)

tuxiang lijie xitong

图像理解系统 (image understanding systems)

由图像处理模块、三维物体模型库及相应的匹配与识别算法所组成的用以对摄取的二维图像进行解释的计算机视觉系统。

解释就是理解。解释的内容则是随着输入图像和理解的目的之不同而不同。例如,对于一张机场的航空照片,重要的也许是机场上有多少架飞机,是什么型号的等等,而对一张 X 光胸片,需要的解释也许是它是否正常,如不正常,可能是什么病等等。一个解释通常要包括识别出图像中有多少感兴趣的物体,它们是些什么物体以及它们之间有些什么关系等。图像解释系统则是能实现上述这种图像理解的计算机系统。它的主要功能是:

(1) 从二维的图像中提取出有意义的二维特征和它们的属性,例如区域、边缘、光流场以及它们的

位置、大小、方向等属性。

(2) 根据一定的知识和它们的属性,将上述这些二维特征分割成一些组,使每一组特征对应于一个三维物体或其组成部分,并提取出后者的属性,例如物体表面的类型,它的边界等等。

(3) 将从二维图像中获得的物体及其属性与已存放在系统中的物体模型进行比较匹配,也就是识别和理解。当一个个物体全部被识别后,就可以完成对整个图像的解释了。

由于物体是三维的,而图像是二维的,三维物体的二维图像随着成像时的角度和光照条件的不同以及噪声等影响,使上述理解过程相当困难。而且在一般情况下,场景中往往有多个物体,前面的物体挡住后面的物体,更给解释工作增加了困难。因此,整个理解过程并不能像上面所说的那样从二维图像到三维场景一次完成。通常,都要经过若干次自底(二维图像)向上(三维场景)和自顶向下的多次反复才能得到较为满意的结果。

自 70 年代后期以来,许多研究工作者,针对某些特定的应用领域(航空照片、室外场景、室内场景、医学图像等),研制了不少图像理解系统,下面是一些典型的例子。

(1) 由美国 Stanford 大学研制的 ACRONYM 系统,用于理解机场的航空照片,能识别出飞机、建筑物、道路、树木等。

(2) 由美国 Carnegie-Mellon 大学研制的 SPAM 系统,也是用于理解机场的航空照片的。

(3) 由美国 Massachusetts 大学研制的 VISION 系统,用于识别室外的普通照片,能识别出房屋(包括分出其主要成分:墙壁、屋顶、门窗等)、汽车、草地、树木、天空等。

(4) 由美国 Stanford 国际研究院研制的 ICS 系统,用于理解室内的照片,能识别出门窗、墙壁以及桌子、椅子等办公用品。

(5) 由加拿大多伦多大学研制的 ALVEN 系统,用于理解人的左心室的 X 光电影(即一组连续的 X 光照片),能识别出左心室,测出不同时刻左心室的体积、直径、宽度和长度的变化情况,以及给出某些初步的诊断供医生参考。

以上这些系统,在少量精心选择过的图像上都获得了较好的结果。但总的说来,与人的视觉系统相比,无论在通用性、可靠性(稳健性),还是在处理速度等方面都还有很大的差距,需要进行大量的研究工作,才能研制出一个真正通用的图像理解系统。

参考文献

1. Hanson A, Riesman E. Computer Vision Systems. New York: Academic Press, 1978
2. Binford T. Survey of Model-based Image Analysis Systems. International Journal of Robot. Res. Vol. 1 (1), 1982, 18 ~ 64 (吴立德)

tuxiang linyu yunsuan

图像邻域运算 (image neighborhood operation) 以图像中某一像素为中心, 利用该像素及其邻域像素的灰度进行的运算。运算的结果将赋值于中心位置的像素。邻域运算的方法有空间域中的线性滤波(卷积)、非线性滤波(选择性平滑、中值滤波等)及图像相关运算等。典型的邻域运算是图像平滑和图像锐化。

模板卷积 模板卷积是一种非常有用的空域线性滤波的基本方法, 可用于增强图像中的某种特征, 削弱另外一些特征, 检测边界, 平滑噪声等。模板是一个具有特定值的阵列, 当一个模板在图像中移动、相乘、求和, 就完成了所谓的“卷积”操作。设 $T(i, j)$ 是一个 $m \times n$ 模板 (m, n 是正整数), $f(x, y)$ 是一幅 $M \times N$ 的图像, 则 T 和 f 的卷积可写为

$$g(x, y) = T * f(x, y) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} T(i, j) f(x-i, y-j) \quad (1)$$

在一些图像处理的文献及应用中, 所谓的“卷积”往往表示的是互相关计算, 即

$$g(x, y) = T \cdot f(x, y) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} T(i, j) f(x+i, y+j) \quad (2)$$

这样做只是处理上的方便, 因为利用式(1)进行计算时, 需要对模板进行“反折”。

非线性滤波 有几种非卷积模板的图像邻域运算是自适应的图像非线性滤波。包括中值滤波和基于图像局部统计的选择平均法和加权平均法。这类方法是根据图像窗口内的图像内容进行的邻域运算, 目的在于平滑图像的同时保护边缘。

中值滤波 是一种经典的非线性低通滤波方法。以某点为中心的窗口内的像素值按从大到小排列, 其中间值(当窗口内元素为偶数时可取两中间值之平均)作为该点处的输出结果。中值滤波有利于除去孤立点或线的噪声, 同时保持了图像的空间分辨率。但不容易去除高斯噪声。

选择平均法 只对灰度值相同或相近的像素进行平均, 以免造成目标边缘的模糊。选择平均法的模板是随图像可变的模板, 系数非 0 即 1, 模板大小一般取 3×3 , 如果模板大, 去噪声效果虽明显, 但计算复杂, 又容易抹去图像细节, 造成模糊。

加权平均法 按照邻域像素灰度特殊的程度加权之后再求和, 以免造成目标边缘模糊。

图像相关运算 相关操作可用于判别在一幅图像中是否存在某一已知的形状。对于一个 $M \times N$ 的图像 $I(x, y)$ 可用已知形状的模板 $t(i, j)$ 通过相关运算, 从图像中找出该形状, 其相关定义是

$$C(x, y) = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} t(i, j) I(x+i, y+j) \quad (3)$$

其中模板的大小为 $m \times n$ 。使用相关运算后, 在图像中和模板相匹配的位置相关函数 $C(x, y)$ 将出现峰值。即表示该位置上存在已知形状。

图像平滑 图像平滑用来减少图像中的噪声。图像平滑技术包括空域中的邻域平均法、中值滤波法和加权平均法、多图像平均法和频域中的低通滤波法(参见图像变换运算)。

图像锐化 图像锐化通过增强图像中的高频成分, 加重图像轮廓以减少图像的模糊。在空域中, 图像锐化要通过微分运算来进行。微分运算可以发现图像中灰度的变化, 但不能保持图像的灰度等级。所以, 基于微分的高通滤波适于边缘检测(参见图像边缘检测), 但不一定适合一般的图像增强的要求。为了达到图像增强的目的, 可以利用锐化滤波, 即把微分滤波器和原图像组合起来。图像锐化也可用频域中的高通滤波来实现(参见图像变换运算及图像变换)。

参考文献

1. 余松煜, 周源华, 吴时光. 数字图像处理. 北京: 电子工业出版社, 1989
2. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989 (朱志刚)

tuxiang moxing

图像模型 (image model) 图像的物理或数学表达及其在计算机中进行储存和加工时所采用的表示。

在图像处理中, 把人眼能够看到的客观世界称为景物。当从某一点观察景物时, 物体所发出的光线进入人的眼睛(或摄像机), 在人眼的视网膜上(或摄像机的传感器上)成像, 该像反映了客观景物

的亮度和颜色随空间位置和方向的变化,因此它是空间坐标的函数。所谓**图像**,就是视觉景物的某种形式的表示和记录。我们都熟悉彩色或黑白照片。黑白(或称单色)照片记录了景物中(物体表面)每点的亮度,而彩色照片不仅记录亮度还记录颜色。随着科学技术的发展,人类不仅能够获得并记录那些人眼可见的图像信息,并且可以利用非可见光和其他手段成像,并转换成人眼可见的图像,如X射线成像、红外线成像、超声成像、核磁共振成像、激光距离成像等(参见**图像获取**)。这些成像手段使得人的视觉能力大大地增强和延伸。

图像模型 人眼看到的景物所对应的一幅平面图像上的信息首先表现为光的强度。它是随位置坐标 (x, y) ,光线的波长 λ 和时间 t 而变化的,因此**图像函数**可以写成:

$$I = f(x, y, \lambda, t) \quad (1)$$

如果只考虑不同波长光的能量作用,那么在视觉效果上只有黑白深浅之分,而无彩色变化。这样的图像被称为**黑白图像(灰度图像)**。灰度图像模型可表示为:

$$I(x, y, t) = \int f(x, y, \lambda, t) V_s(\lambda) d\lambda \quad (2)$$

其中 $V_s(\lambda)$ 为相对视敏函数。

当考虑不同波长光的彩色效应时,就得到**彩色图像**。根据三基色的原理,任何一种彩色都可分解为红、绿、蓝(RGB)三种基色,所以彩色图像可表示成

$$I = \{R(x, y, t), G(x, y, t), B(x, y, t)\} \quad (3)$$

式中

$$R(x, y, t) = \int f(x, y, \lambda, t) R_s(\lambda) d\lambda$$

$$G(x, y, t) = \int f(x, y, \lambda, t) G_s(\lambda) d\lambda$$

$$B(x, y, t) = \int f(x, y, \lambda, t) B_s(\lambda) d\lambda$$

其中 $R_s(\lambda)$, $G_s(\lambda)$ 和 $B_s(\lambda)$ 分别是R、G、B三种基色的视敏函数。

当图像内容随时间变化时,它们被称为**时变图像或运动图像**。运动物体的图像、电影、电视等都是运动图像。当图像内容不随时间变化时,称为**静止图像**。对黑白静止图像而言,其函数为:

$$I = f(x, y) \quad (4)$$

由于人眼和其他成像系统视野有限,因此图像在空间上是有界的。其取值区域通常定义为矩形。图像函数在某一点的值通常称为亮度,一般用正实

数表示。亮度或灰度的数值是有限的,即

$$0 \leq f(x, y) \leq B_m$$

其中 B_m 为最大亮度。

采样和量化 只有将上面定义的图像转换成**数字图像**,才能用数字计算机进行处理。采样是图像 $f(x, y)$ 进入计算机的第一步。采样过程包括将一个 $(M \times N)$ 的采样网格覆盖在图像 $f(x, y)$ 上,然后度量每个方格的平均亮度,这样,图像 $f(x, y)$ 就可以用计算机中的一个 $M \times N$ 矩阵 $[f(m, n)]$ 来表示。 $f(m, n)$ 为矩阵中一个元素。

采样中的主要问题是采样密度应该多大,才不致丢失原图的信息。采样定理回答了这个问题。对于在 x 和 y 方向上截止频率分别为 ξ_x 和 ξ_y 的有限带宽图像,在 x 和 y 方向上的采样间隔应满足:

$$\Delta x \leq 1/(2\xi_x), \quad \Delta y \leq 1/(2\xi_y) \quad (5)$$

如果 $f(x, y)$ 在 $x \in [0, X]$, $y \in [0, Y]$ 内有定义,并以 Δx 和 Δy 为间隔取样,则沿 x 和 y 方向的取样点数为: $M = X/\Delta x$, $N = Y/\Delta y$ 。

对图像 $f(x, y)$ 的采样完成后,得到图像各点的采样值 $f(m, n)$ 。但是,在输入数字计算机之前,对 $f(m, n)$ 还要进行量化。具体地说,就是要在样本值的取值范围内进行分层(分成 k 个区间),然后用单个值(逻辑值)来代表各层内所有可能的亮度(物理值)。根据计算机内整数存放的惯例,一般可以把样本的取值分成 $k = 2^l$ 个层次,例如 $l = 6, 7$ 或 8 ,即把矩阵中每个元素的灰度值分成64, 128或256个层次。一般来讲,层次越多,由量化的取值恢复的实际图像(如显示的图像)越接近原图,看上去越满意。当层次不够时,在图像中缓慢变化的区域将出现原图像所没有的假轮廓。

最简单的量化方案是**均匀量化**,即把样本值的整个取值范围均分成 k 个子区间。如果样本值在某个取值范围内频繁出现,而在其他范围内很少出现,就可以在样本值频繁出现的范围内采用小量化区间的密集量化,而在其他地方量化得粗糙一些,这样可以在不增加量化层次的条件下,减少实际引入的量化噪声,这就是**非均匀量化**。

亮度和对比度 在均匀量化中,假如量化级为256,则0代表最小亮度(黑),255代表最大亮度(白)。亮度值在0~255范围内呈线性关系。例如,灰度值为42的点是灰度值为21点的亮度的两倍。图像**对比度**反映了图像中亮度的动态范围。如在256级灰度下,一幅具有灰度范围 $[50, 100]$ 的图像就比一幅灰度范围为 $[30, 210]$ 的图像的对比

度低。

像素和分辨率 尺寸为 $M \times N$ 的数字图像矩阵 $[f(m, n)]$ 中的每个元素称为图像元素, 简称**像素**, M 表示水平行数, N 表示垂直列数。像素是数字图像可以赋值的最小单位。每个像素对应的图像物理尺寸由采样决定。术语“**分辨率**”描述了数字图像所能表示的空间细节的程度。图像分辨率一般指数字图像中像素的个数 ($M \times N$)。对分辨率的要求常常是和应用有关的。例如, 为了产生满意的电视效果, 分辨率 ($M \times N$ 的大小) 应不少于 500×500 。一般计算机显示的分辨率为 640×480 。当然, 有的应用需要较高的分辨率为 1024×1024 , 而另一些只要较低的分辨率为 32×32 就够了。图像获取和图像显现中有关分辨率的定义和具体设备相关。

数字图像的表示 针对图像的不同含义或不同的处理要求, 数字图像常用的表示形式有:

灰度图像 (或称单色图像) 每个像素用一个字节表示, 256 级灰度可表示高质量的图像。而且表示方法也比较紧凑 (八位二进制数即一个字节)。512 \times 512 \times 8 位的图像和标准电视分辨率相近, 但有更好的灰度层次。有的数字图像特别是医学图像, 需要用十六位二进制数表示, 而且常常是有正负号的。这种有符号的表示对一些图像处理的中间结果也非常有用, 它可以避免数据的上溢和下溢。

浮点数图像 实数可以用作图像处理的中间结果, 它既可以保留运算中产生的小数部分, 又可以表示很大的数据范围。复数也可用于图像的表示, 主要用于数字图像转换到其他域中的表示, 如傅里叶变换域。

彩色图像 彩色图像的采样和单色图像相似, 但需要分别测量和采样红、绿、蓝三基色分量, 因此彩色图像的数据量是单色图像的三倍。在实际应用中, 彩色图像几乎总是数字化成各为八位的三个分量。这种形式亦称为 24 位彩色图像, 意即每个像素需 24 位表示; 有时也说这种图像可以表示出 $2^{24} \approx 16 \text{ M}$ (兆) 种颜色。

二值图像 在每个像素上只表示了某种特性的有或无, 即逻辑表示上的真或假, 称作二值图像。通常二值图像的每个像素只用 1 个二进制位 (bit) 表示, 一个字节表示 8 个像素 (一般按行储存)。在实际中, 图像的一行像素用整数个字节表示, 典型的应用有文本图像处理、图像打印以及图像表示等。

参考文献

1. 余松煜, 周源华, 吴时光. 数字图像处理. 北

京: 电子工业出版社, 1989

2. Lewis R. Practical Digital Image Processing. Ellis Horwood, 1990 (朱志刚)

tuxiang quyue biaoshi

图像区域表示 (image region representation) 把图像中的某些区域用合适的数据结构或数学式表示的方法。图像的区域表示方法有 y 轴表示及四叉树表示等。

y 轴表示 这是一种将区域置于 x - y 平面内并用相应的坐标排列来表示区域的方法。 y 轴表示是一个含有多个子表的表格。主表的数字对应于区域元素所在的 y 坐标, 子表的第一个数字是同一 y 坐标下第一个区域元素的 x 坐标, 子表的第二个数字是该区域最后那个元素的 x 坐标。若该区域的右面还有别的区域, 子表依上述规则继续排列。图 1 是一个用 y 轴表示的例子。图中的区域可表示为 $[(3, 4, 5)(2, 5)(3, 4)(2, 3, 4, 5)]$ 。

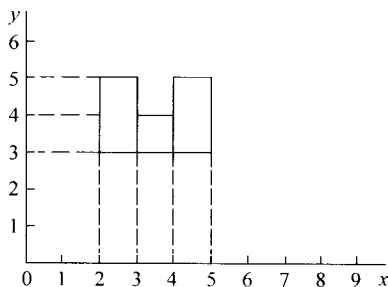


图 1 y 轴表示

四叉树表示 这是对图像所在区域用树形结构表示的一种方法。这种方法在数字图像处理、景物理解以及计算机图形学等有广泛的用途。

四叉树是对图像所在平面进行递归划分形成的。首先以包围整幅图像的一个矩形为根结点, 然后将该矩形作 4 等分, 形成 4 个子结点, 并加以编号。如果某一结点对应的矩形全部被图像所充满, 则称为黑结点; 如果某一结点对应的矩形无图像存在, 称为白结点; 如果某一结点对应的矩形部分地被图像占有, 则称为灰结点。黑白两种结点不再细分; 而灰结点则需作进一步划分。这一过程递归进行, 直至全部为黑白两种结点或者达到预先规定的精度为止。

应该指出, 四叉树是图像所在区域的表示。一幅图像及其四叉树表示见图 2。

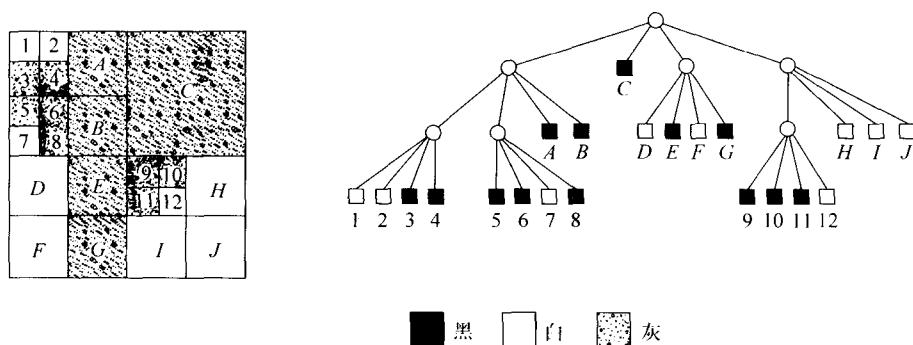


图2 四叉树表示

参考文献

1. 赵荣椿等. 数字图像处理导论(第二版). 西安: 西北工业大学出版社, 1995
2. Sonka M, Hlavac V, Boyle R. Image Processing, Analysis and Machine Vision. Chapman & Hall, 1993 (曾建超)

tuxiang quyu fenge

图像区域分割 (image region segmentation)

基于图像区域特性的差异对图像进行分割的技术。区域分割的基本思想是标识图像中各个具有相似特征的区域。相似的特征可以是形状、像素值或纹理等。在模式识别中的聚类技术也可用于基于区域的图像分割。

模板匹配 基于区域分割图像的一种直接方法是将图像中的区域和一组给定的模板进行比较匹配,从而将符合模板的物体从图像的其他部分中分割出来,而剩余的图像则可根据需要再用其他方法分析。例如,模板匹配可用于分割图文混排的书稿。当文字用模板匹配的方法找出来以后,图形可再用其他方法进行分析。模板匹配的过程往往用相关或卷积计算来进行(参见图像处理的基本运算)。

纹理分割 当物体置于明显的纹理背景中或物体本身具有较强的纹理特征时,就需要利用基于纹理的区域分割方法。由于纹理是某种模式,或者说图案、花样、结构等的重复,所以不能用单个的像素的特性(灰度或颜色)来描述。当然也无法用基于像素的分类方法(参见图像像素分类)。由于纹理经常包含大量的边缘,因此,除非滤去纹理,否则用边界跟踪的方法分割有丰富纹理的图像很难有好的效果。

纹理的描述与分类是分割的基础(参见图像特征提取)。当知道图像中有某种纹理存在时,可利用已知纹理的特征(如该纹理在频域中的描述或空间灰度关系矩阵)在图像中寻找。如果事先没有知识的话,可以采用基于区域的聚类方法进行纹理区域的分割。一种容易想到的办法是:把图像分成若干(小)块,计算每一块的纹理特征,根据特征差别的程度决定是否把小块合并。

区域聚类法 聚类法一般可分为区域生长法及分裂合并法。

区域生长 区域生长的基本思路是:从满足检测准则的点或一块区域开始,在各个方向上“生长”物体。“生长”的依据是:同一类型区域的特征,如灰度、颜色及纹理特征等,相差不会太远。满足一定合并条件的邻域可以并入该区域。在生长过程中,合并条件可以调整。当再也找不到可合并的邻域时,生长停止。

区域的分裂和合并 这个方法的基本思路是:首先将图像分为若干“初始”区域,然后再分裂或合并这些区域,逐步改进区域分割的指标,直到最后将图像分割为数量最少(或符合某一要求)的“基本一致”的区域为止。通常,“一致”性的标准可用特性的均方误差来量度。

与基于边界的图像分割方法(参见图像边缘检测)相比,基于区域生长法和分裂合并法对噪声相对不敏感,但是计算复杂度较高。(俞志和)

tuxiang tezheng tiqu

图像特征提取 (image feature extraction)

检测或获取图像中特征的过程和技术。在计算机图像处理及机器人视觉中,特征一般为点、线、边缘、纹

理以及一些特定的形状等。

图像特征的提取对于**图像分割**及图像理解都有很重要的意义。例如,检测到的边缘往往被认为是图像中不同物体或同一物体不同表面的分界。特征的分布及形状直接影响到进一步的图像处理及理解。

一般来讲,特征是由图像中的不连续或者说空间(即二维图像平面)中的亮度(或颜色分量)不光滑过渡而形成的。所以,特征提取也就是检测图像中的不连续特性的过程。而这些不连续特性一般是表现在点、线和边缘。

点的检测 点的检测或提取往往用来去除图像中的噪声或进行图像中的颗粒分析等。孤立点的检测可以用一个简单的模板,例如下面的 3×3 模板来进行。

$$\begin{bmatrix} W_1 & W_2 & W_3 \\ W_4 & W_5 & W_6 \\ W_7 & W_8 & W_9 \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

在需要检测的图像区域中,每一个 3×3 的小块与上面的模板中的相应数值相乘再相加(参见**图像邻域运算**)。假定图像中的某一块是

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix}$$

检测的结果就是

$$R = x_1 W_1 + x_2 W_2 + \cdots + x_9 W_9 \\ = -x_1 - x_2 - \cdots + 8x_5 - \cdots - x_9$$

很明显,如果图像中的 $x_1 \sim x_9$ 对应于一块均匀的区域,则 $R=0$ 。如果 $R \neq 0$,那么被检测的 3×3 区域里必然不均匀。由于模板的中间的元素 x_5 取值 8,如果它与其周围的点有区别的话,差别在计算结果 R 中就会被明显地反映出来。为了检测 x_5 是否是一个特殊点,需要确定某一门限值 T (参见**图像像素分类**),当 $R > T$ 时,可认为 x_5 是一特殊点。换句话说,这个特征点被检测了出来。

线的检测 线的检测可以用下面所示类型的模板。通过模板匹配(参见**图像区域分割**)的方法来进行。

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}, \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}, \\ \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}, \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}.$$

很明显,它们可以用来检测水平线,垂直线以及 45° 度斜线。

边缘检测 简单的边缘检测可以用 2×2 的模板来进行。它们是

$$\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \text{ 和 } \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}.$$

也可以用 3×3 的模板

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}.$$

其他类似的模板还有

$$\begin{bmatrix} -1 & 0 & 1 \\ -\sqrt{2} & 0 & \sqrt{2} \\ -1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{bmatrix}, \\ \begin{bmatrix} \sqrt{2} & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -\sqrt{2} \end{bmatrix}, \begin{bmatrix} 0 & 1 & \sqrt{2} \\ -1 & 0 & 1 \\ -\sqrt{2} & -1 & 0 \end{bmatrix}.$$

上面介绍的特征提取模板技术是对像素而言的(参见**图像边缘检测**)。在实际的图像中,由于噪声的关系,不可能只采用上面的检测技术用孤立的像素点来形成完整的直线或边界。进一步的处理是必要的。例如,为了确定检测出来的像素是否在一条直线上可以用霍夫变换。

霍夫变换用于检测图像中某些像素之间的结构关系。例如,为了确定图像中的若干个点 $(x_1, y_1), (x_2, y_2), \dots, (x_k, y_k)$ 是否在某条直线上,可以用霍夫变换。

这种变换可以通过极坐标来进行,其表达式是

$$\rho = x \cos \theta + y \sin \theta$$

在 x, y 坐标系中,所有通过某一个点的直线在 ρ, θ 的极坐标系中对应一条正弦曲线。所以,在 ρ, θ 极坐标中若干条正弦曲线的交点就代表了若干个点在 x, y 坐标系中共线(落在一条直线上)。例如,在 x, y 坐标系中有 A, B, C, D, E 五点。通过 A 的所有的直线在 ρ, θ 极坐标系中构成了正弦曲线#1。类似地, B 相当于正弦曲线#2, C —#3, D —#4, E —#5。如果发现在 ρ, θ 极坐标系中,正弦曲线#2, #4, #5 交于一点,那么,就知道 B, D, E 在 x, y 坐标系中共线。

霍夫变换也不限于直线。一般来讲,如果一个函数可写成 $g(\mathbf{x}, \mathbf{c}) = 0$ 的形式,其中 \mathbf{x} 是坐标向量, \mathbf{c} 是参数向量,都可以进行变换。例如,圆的方程是

$$(x - c_1)^2 + (y - c_2)^2 = c_3^2$$

可以进行变换,在 c_1, c_2, c_3 的坐标中(称作参数空

间)来表示圆。当然,变换的复杂程度以及变换所带来的好处取决于具体的问题及变换方法的选择。

纹理检测 纹理是某种相似的模式,或者说图案、花样、结构等的重复。通常把一个单独的图案称为纹理元素。由于纹理元素是由多个像素构成的,所以不能只根据单个像素的特性(灰度或颜色)来确定纹理,当然也无法用上面介绍的特征提取模板及霍夫变换等检测方法。

纹理的检测可以在频域内或空间域内进行。频域中的方法是:对图像(或分块)进行傅里叶变换(参见**图像变换**),分析其频谱能量分布。通常将低频部分忽略,而寻找高频部分的能量尖峰,确定它们的波长(频率)及方向。能量尖峰(可能有几个)的频率及方向代表了这种纹理的特征。

空间灰度关系矩阵也可以用来在图像中确定纹理。空间灰度关系矩阵是这样定义的:矩阵 $S_{d,\theta}(i,j)$ 中下标 d, θ 的意义是两个像素之间的距离(d)及相对位置角度(θ)。一般 d 可以是一个,两个或多个像素(距离),而角度 θ 一般为 $0^\circ, 45^\circ, 90^\circ, 135^\circ, 180^\circ, 225^\circ, 270^\circ, 315^\circ$ 。矩阵 $S_{d,\theta}(i,j)$ 在 (i,j) 位置的值等于图像中灰度是 i 和 j 的像素对的数目,这些像素对的相对距离是 d 、相对角度是 θ 。用下面的示意图进一步说明 $S_{d,\theta}(i,j)$ 的算法。例如, $d=1, \theta=0$ 有元素 $S(1,1)=0; S(1,2)=6; S(2,1)=6; S(2,2)=0$ 。而 $d=1, \theta=45^\circ$ 时,则 $S(1,1)=5; S(1,2)=0; S(2,1)=0; S(2,2)=4$; 如此等等。

2	1	2	1
1	2	1	2
2	1	2	1
1	2	1	2

利用空间灰度关系矩阵,可以计算“能量”

$$\sum_i \sum_j [S_{d,\theta}(i,j)]^2$$

能量大的 $S_{d,\theta}(i,j)$ 表示图像上以 d, θ 关系重复的纹理较多。也可以计算所谓的“惯量”或“对比度”

$$\sum_i \sum_j (i-j)^2 S_{d,\theta}(i,j)$$

如果图像的灰度没有变化,则对比度为零。如果图像是黑白间隔(一条白一条黑)的,那么 $d=1$ 的空间灰度关系有较大的对比度。

参考文献

1. 赵荣椿等. 数字图像处理导论(第二版). 西安: 西北工业大学出版社, 1995

2. Jain A K. Fundamentals of Digital Image Processing. Prentice-Hall Inc., 1989 (俞志和)

tuxiang tuihua

图像退化 (image degradation) 在成像过程中由于成像系统本身或噪声等多种因素的影响使图像变得模糊的现象。图像退化也称为劣化。分析和了解退化现象及其原因,建立退化过程的数学模型是进行图像复原的必要条件。基于退化的模型,我们可以运用相反运算去掉模糊,改进图像的质量。图1中表示了几种退化现象的图例。在图1中,(a)是在摄影时由于曝光量和感光密度的非线性引起的非线性退化;(b)是由于散焦或光的衍射作用引起的退化;(c)是曝光期间摄像机与物体之间有相对运动而产生的运动模糊;(d)是叠加有随机噪声的模糊图像。(a)、(b)和(c)的情况是与成像系统和成像过程有关的。(d)的情况可能与成像过程无关,如相纸上的斑点、显示噪声等。图2将退化过程综合为一个模型,与其对应的数学表达式为:

$$f(x,y) = H[g(x,y)] + n(x,y) \quad (1)$$

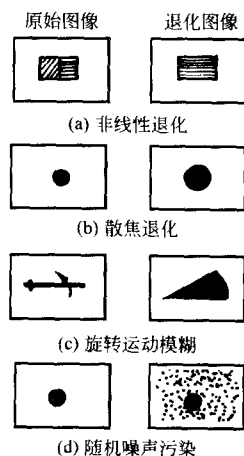


图1 几种退化现象的图例

图2的模型表示,一幅模糊图像 $f(x,y)$ 可以看作是原始图像 $g(x,y)$ 经过算子 H (这里把图像的退化过程模型化为一个算子,或者说施加于图像 $g(x,y)$ 的一种运算)作用后再叠加噪声 $n(x,y)$ 的结果。

对于图1中(a)的情况,算子 H 是非线性函数。对于图1中(b)的情况,算子 H 具有线性和位移不变特性,即输出图像只决定于图像点的幅值,与图像

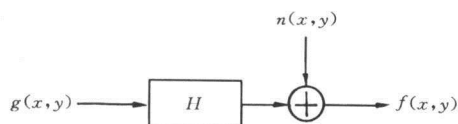


图2 图像退化过程的模型

点在空间的位置无关。对于图1中的(c),算子 H 由摄像机与物体之间的相对运动来决定。对于图1中(d)的情况,通常假定噪声 $n(x, y)$ 是随机噪声,并且具有有限幅值。实际退化图像经常是其中几种退化现象的综合结果。

参考文献

赵荣椿等. 数字图像处理导论(第二版). 西安: 西北工业大学出版社, 1995 (李树青)

tuxiang weina lübo fuyuan

图像维纳滤波复原 (image restoration by wiener filtering)

满足最小均方误差准则的一种图像复原方法。由于大多数退化图像都含有噪声,使用反向滤波法(参见图像反向滤波复原)经常导致复原失败。针对这种情况,可以应用维纳滤波。假定退化模型为线性并且空间位移不变。原始图像 $g(x, y)$ 、退化图像 $f(x, y)$ 均为平稳随机函数, $n(x, y)$ 为幅值有限的加法噪声,点扩展函数 $h(x, y)$ 是已经确知的函数,那么可求得复原图像的估值 $\hat{g}(x, y)$ (参见图像反向滤波复原),使其与原始图像 $g(x, y)$ 之间的均方误差 e^2 最小:

$$e^2 = E[|g(x, y) - \hat{g}(x, y)|^2]$$

这种满足均方误差最小的复原方法就是维纳滤波。

维纳滤波的复原公式如下(这是最常用的维纳滤波的频域解):

$$\hat{G}(u, v) = \frac{1}{H(u, v)} \times \frac{|H(u, v)|^2}{|H(u, v)|^2 + r S_n(u, v) / S_g(u, v)} F(u, v) \quad (1)$$

$u, v = 0, 1, \dots, M-1$

式中, $H(u, v)$ ——点扩展函数的傅里叶变换;

$F(u, v)$ ——退化图像的傅里叶变换;

$\hat{G}(u, v)$ ——复原图像估值的傅里叶变换;

$S_g(u, v)$ ——图像功率谱(图像自

相关矩阵的傅里叶变换);

$S_n(u, v)$ ——噪声功率谱(噪声自相关矩阵的傅里叶变换)。

这里 $S_g(u, v)$ 和 $S_n(u, v)$ 是从属于某一类的许多幅图像统计而得的参数,而 r 是供选择的参数。

当 $r=0$ 时,式1变为

$$\hat{G}(u, v) = \frac{F(u, v)}{H(u, v)} \quad (2)$$

这就是反向滤波。

当 $r=1$ 时,

$$\hat{G}(u, v) = \frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_n(u, v) / S_g(u, v)} F(u, v) \quad (3)$$

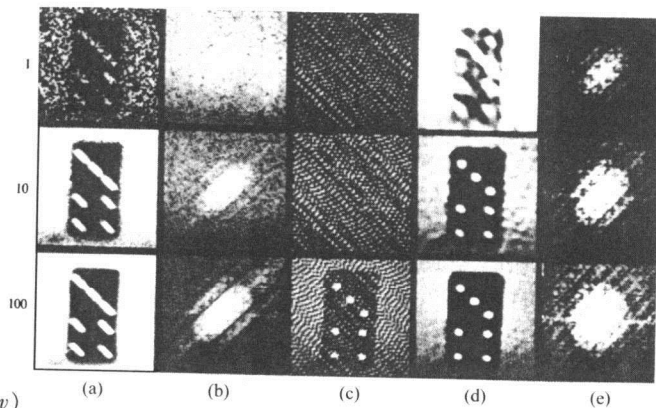
$u, v = 0, 1, \dots, M-1$

这种情况称之为标准维纳滤波。

r 也可以取其他值。通过调整 r 可获得不同的

原始图像的近似值 $\hat{G}(u, v)$,以满足均方误差减少的条件。这称之为带参数的维纳滤波。

当采用维纳滤波时,信号与噪声的比对复原图像影响比较小(即无论信噪比大或小,复原效果都较好)。图1给出了用维纳滤波和反向滤波对模糊图像进行复原处理的例子。可以看到,当噪声比较大时,维纳滤波复原的图像质量优于反向滤波复原的图像质量。



(a) 退化图像(由上到下信噪比为 1, 10, 100);
(b) 退化图像的傅里叶频谱;
(c) 反向滤波复原的图像;
(d) 维纳滤波复原的图像;
(e) 维纳滤波复原后图像(d)的傅里叶频谱

图1 维纳滤波和反向滤波对模糊图像进行图像复原的比较

(此图取自参考文献3)

参考文献

1. 赵荣椿等. 数字图像处理导论(第二版). 西安: 西北工业大学出版社, 1995
2. Jain A K. Fundamentals of Digital Image Processing. Prentice-Hall Inc., 1989
3. Rafael C. Gonzalez / Paul Wintz. Digital Image Processing (second edition). Addison-Wesley, 1987
(李树青)

tuxiang xihua

图像细化 (image thinning) 根据某些限制条件逐步去除图像中某一区域边界像素点的过程和方法。一些典型的限制条件包括: ①不能去除区域的端点或顶点; ②不能破坏区域的连通性; ③不能过度地腐蚀某些区域等。由于细化可以获得保持区域结构特性的骨架(参见**图像骨架表示**), 所以在字符识别, 印刷电路板检测等方面有广泛的应用。

可以采用非循环算法与循环算法来进行细化。典型的非循环算法是中轴变换(参见**图像骨架表示**)。该算法寻找并保留区域的对称轴线以完成细化过程。循环算法常用的是**图像形态学运算**。

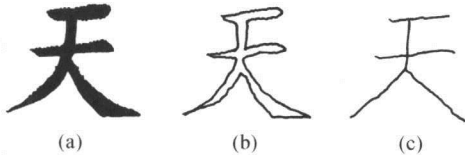


图1 “天”字的边界及细化

用一个例子来说明细化不同于提取边界的应用。图1(a)是一个毛笔大字“天”。如果为了研究其书写风格或产生一个计算机中的毛笔字, 可能要确定笔划的粗细、形状。那么在计算机处理中应该提取边界并存储边界的形状, 如图1(b)所示。如果为了让计算机识别这个字, 就可能是要细化(收缩)区域, 以获得骨架, 如图1(c)所示, 以便进一步处理。

(俞志和)

tuxiang xiangsu fenlei

图像像素分类 (image pixel classification)

根据图像像素特征的差异对像素进行分类的过程和技术。分类的目的是为了从一幅图像上找出所关心的物体。所说的图像像素特征可以是颜色、亮度或其他像素的特征。我们统一称之为图像像素值。

直方图

图1是一幅校园道路图及其亮度的直方图。直方图的横坐标是灰度值, 纵坐标是亮度在某一范围内的像素的数目。例如, 直方图中可以看出, 亮度值在192到224范围内的像素比较多。从直方图上还可以看出, 图像的亮度的分布有两个峰。对比直方图与校园道路图可以发现直方图上亮度较大的那个峰对应的是道路区域中的那些像素。非道路区域比较暗, 所以非道路区域的像素落在直方图上亮度较低的那个峰附近。我们如果在两个峰之间的谷将直方图中的像素分开, 结果也就是在校园图上把道路和非道路进行了分割。直方图提供了一个直观的方法帮助我们进行图像分割。确定分割的界线(或称确定“门限”)的过程则是下面要介绍的问题。

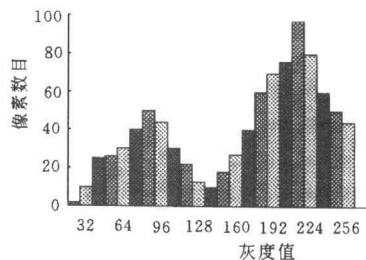
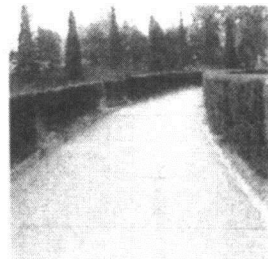


图1 校园道路及亮度直方图

像素值门限化 当像素值足以表示某些物体的特性时, 可利用设置门限的方法对图像进行分割。为了得到较好的效果, 往往通过对图像像素值进行校正或增强(参见**图像增强**)使得一给定的像素值范围能表示某一物体(或有意义的部分)。例如, 在一幅遥感红外图像中, 或许大幅度值代表较高的温度; 通过门限化也许可以分出陆地与水面。门限化也可用于多光谱图像、彩色图像和X光图像等。

在门限化技术中, 门限的选择是关键性的一步。常用的方法有: ①寻找直方图的峰、谷点, 在多峰情况下, 一般可选谷点作为门限值。②选择某一门限值, 使得事先规定的一定百分比的像素点落在门限

值内。③如果不同类的统计模型已知(如高斯分布),可通过求最小概率误差或其他度量(如贝叶斯风险)来确定门限值(参见模式识别)。④调整局部的直方图再确定门限。

部件标记 一种对二值图像简单却有效的分割方法是标记法,即通过检查像素与其邻域的连通性,并将连通部分标记为一个区域(集合)。不同的标记即为不同的物体,像素标记是这样进行的:假定按由左到右,由上到下的顺序扫描一幅二值图像,将连通的像素用记号标记为属于某一物体(或与其连通的物体)或背景。标记的过程中可以用“四连通法”或“八连通法”(当一个像素与其相邻的4个或8个像素之一有相同的性质时,就称这个像素是四连通或八连通的),举例如下:

1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72

图2 利用标记法进行图像分割

图2给出了一个例子。图上用粗体数字代表黑色的像素,其他的数字代表白色的像素。从左到右看第一行,相连的2、3、4、5为第一号物体,9、10、11为第二号物体。第二行,15与3连通,16与15及4连通等等,所以15、16、17属于第一号物体。类似地,21、22、23属于第二号物体,如此等等。那么,42号像素是否与29号像素连通呢?这就要根据具体情况来规定了。如果对角线方向上也可以算连通的话,这种规定就称作八连通,否则在四连通的情况下,它们就不连通了。(俞志和)

tuxiang xingtaixue yunsuan

图像形态学运算 (image morphological operation)

基于数学形态学的图像运算。它通过图像中被处理对象和结构元素的相互作用来获取该对象(物体)的拓扑或结构信息。数学形态学是研究形状和结构的学科。在图像处理中,形态学运算采用不同于空域滤波和频域滤波的形态滤波方法,也就是通过图像中被处理对象和结构元素的相互作用以得到更本质的形态。

在图像处理时,利用形态学基本概念可以改善图像质量。例如,当希望获得人脸的线画图时,由于光照等原因,人脸的边缘会有噪声。使用传统图像分割技术,人脸轮廓可能不光滑。利用形态学方法虽然不能产生新的信息,但可根据人脸轮廓一般是光滑曲线这个先验知识,通过平滑除掉边界的凸起和凹陷。

形态学的概念也可以用来描述和定义图像的各种几何参数和特征,如区域数目、面积、周长、连通度、颗粒度、骨架和方向性等。

形态学运算常用于二值图像。实际上,早期的数学形态学只关心位置而不考虑灰度。现在形态学运算已扩展到灰度图像。

腐蚀和膨胀 大部分形态学运算都定义在腐蚀和膨胀两个基本运算的基础上。在二值图像的形态滤波中,它们分别为“结构差”与“结构和”运算。假定处理对象 X 和结构元素 B 为二维欧氏空间中的集合。它们的基本关系为:

B 包含于 X ,记为 $B \subset X$,见图1(a);

B 击中 X ,即 $B \cap X \neq \emptyset$,记为 $B \uparrow X$,见图1(b);

B 不击中 X , $B \cap X = \emptyset$,记为 $B \subset X^c$, X^c 为 X 的补集,见图1(c)。

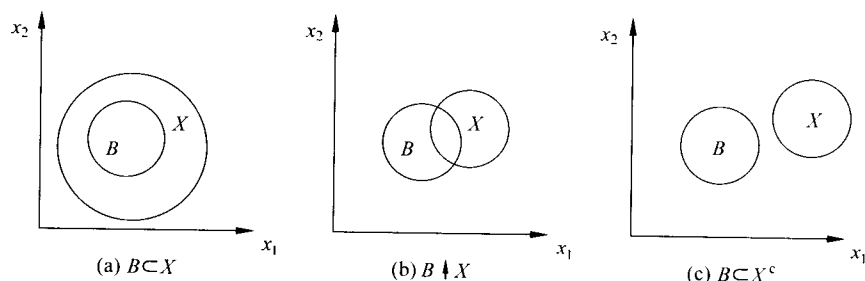


图1 形态学运算的基本关系

把结构元素 B 平移 x (x 是一向量) 以后得到的元素记为 B_x 。例如在图 2 中, 当 B 为 $\{(0,0), (1,0), (0,1)\}$ 而 $x = (2,2)$ 时, B_x 为 $\{(2,2), (3,2), (2,3)\}$ 。

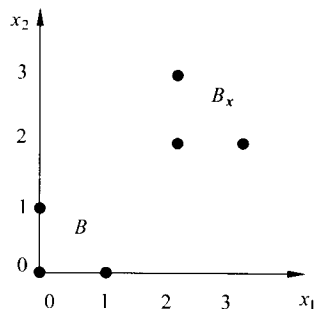


图 2 结构元素的平移

当结构元素被平移 x 后, 使 B_x 包含于 X 的所有的平移量 x 的集合称为 X 被 B 腐蚀的结果, 用公式表示为

$$E(X) = \{x; B_x \subset X\} \text{ 或记为 } X \ominus B_x$$

使 B_x 击中 X 的所有的平移量 x 构成的集合称为 X 被 B 膨胀的结果。用公式表示为

$$D(X) = \{x; B_x \cap X \neq \emptyset\} \text{ 或记为 } X \oplus B_x$$

腐蚀和膨胀互为对偶, 用公式表示为

$$(X \ominus B)^c = X^c \oplus B$$

这里, 上标“c”表示取补集, 即 B 对 X 的腐蚀的补集等于 B 对 X 补集的膨胀。例如, 河岸的补集为河面, 那么对河岸的腐蚀等效于河面的膨胀。在有些情况下这种对偶关系非常有用。例如, 某个图像处理系统用硬件实现了腐蚀, 那么膨胀可以通过先将求反的原图进行腐蚀运算再将结果求反获得, 而不必要再构成膨胀的硬件。

开和闭 在腐蚀和膨胀的基础上, 常用的形态运算(变换)有结构开和结构闭、击中不击中变换、细化和粗化、边界和骨架等。这里仅就结构开和结构闭运算做如下解释。

用结构元素 B 对处理对象 X 先腐蚀后膨胀的结果称为 B 对 X 的开, 即

$$\text{OPEN}(X) = (X \ominus B) \oplus \overset{\vee}{B}, \text{ 记为 } X_B。$$

这里, $\overset{\vee}{B}$ 是 B 的对称集, 即 $\overset{\vee}{B}$ 是把 B 中的元素改变符号(由正变负, 由负变正)的集合。例如, $B = \{(0,0), (1,0), (0,1)\}$, 则 $\overset{\vee}{B} = \{(0,0), (-1,0), (0,-1)\}$ 。

而用结构元素 B 对处理对象 X 先膨胀后腐蚀

的结果称为 B 对 X 的闭, 即

$$\text{CLOSE}(X) = (X \oplus \overset{\vee}{B}) \ominus B, \text{ 记为 } X^B。$$

一般来说, 结构开能去除孤立的小点、毛刺和小桥, 而结构闭则能填平小湖、弥合小裂缝。

灰度图像的形态滤波 形态运算可用于一般的灰度图像。一种简单方法是将灰度图像二值化, 即将所关心的灰度变为 1, 其他变为 0。更好的方案是定义和二值形态运算略有不同的“灰值形态运算”。灰值腐蚀和膨胀分别用于灰度图像区域边界的突起抹平和下凹填补。同二值图像一样, 同样可以在腐蚀和膨胀运算的基础上定义其他运算。

参考文献

1. 余松煜, 周源华, 吴时光. 数字图像处理. 北京: 电子工业出版社, 1989
2. Jain A K. Fundamentals of Digital Image Processing. Prentice Hall Inc., 1989 (朱志刚)

tuxiang yundong mohu fuyuan

图像运动模糊复原 (image motion-blurred restoration)

对由于运动而造成模糊的图像进行恢复的过程和技术。运动模糊图像是由于照相曝光期间照相机与物体之间有相对运动而造成的。例如从飞机或卫星上拍摄地面照片, 或者用照相机拍摄高速运动物体的照片时, 都有可能造成照片(图像)模糊。相对运动是均匀直线运动, 这是经常遇到的典型情况。

假设清晰的图像是 $g(x, y)$, 而造成 $g(x, y)$ 模糊的惟一因素是均匀直线运动所引起的位置移动, 那么在曝光时间为 T 的情况下, 可得模糊图像 $f(x, y)$ 的表达式:

$$f(x, y) = \alpha \int_0^T g\left(x - a \frac{t}{T}, y - b \frac{t}{T}\right) dt \quad (1)$$

这里假定在 T 时间内, 照相机与物体之间的相对运动造成在 x 坐标方向移动了 a , y 方向移动了 b 。式中 α 是和照片感光灵敏度有关的系数。

如果物体只是沿 x 坐标方向由左向右移动(或相机由右向左移动), 将式(1)离散化并把比例系数 α 去掉(令 $\alpha = 1$), 则得到

$$f(x, y) = \sum_{k=0}^{a-1} g(x - k, y) \quad (2)$$

式(2)表示模糊图像中的每一点 x 是由原始图像中的 x 到 $x - (a - 1)$ 共 a 点叠加而成的。

当图像的宽度 $L = Ka$ 时(K 是整数), 可以得到

如下的近似解

$$g(x, y) \approx A - \frac{1}{K} \sum_{k=0}^{K-1} \sum_{j=0}^k f'[x - ma + (k-j)a, y] + \sum_{j=0}^m f'(x - ja, y) \quad (3)$$

这里, $f'(x, y)$ 是 $f(x, y)$ 对 x 的差分, 即

$$f'(x, y) = f(x, y) - f(x-1, y)$$

而 m 是一整数, 取值应满足

$$\frac{x}{a} - 1 \leq m \leq \frac{x}{a}$$

式(3)中的 A 是一常数, 当 K 值较大时 A 趋近 $g(x, y)$ 的平均值, 在实际处理中 A 可以取 $f(x, y)$ 的平均值。

类似地, 当照相机与物体之间的相对运动沿 y 方向移动了 b 时, 复原图像是

$$g(x, y) \approx A - \frac{1}{K} \sum_{k=0}^{K-1} \sum_{j=0}^k f'[x, y - nb + (k-j)b] + \sum_{j=0}^n f'(x, y - jb) \quad (4)$$

其中, $f'(x, y)$ 是 $f(x, y)$ 对 y 的差分, 即

$$f'(x, y) = f(x, y) - f(x, y-1)$$

n 是一整数, 取值应满足

$$\frac{y}{b} - 1 \leq n \leq \frac{y}{b}$$

根据式(3)和式(4), 复原处理可推广到 x, y 方向都有移动的情况。图1给出了一个利用上述方法对运动模糊图像进行复原的例子。



(a) 模糊图像

(b) 复原的图像

图1 均匀直线运动造成的模糊图像的复原
(此图取自参考文献2)

参考文献

1. 余松煜, 周源华, 吴时光. 数字图像处理. 北京: 电子工业出版社, 1989
2. Rafael C, Gonzalez/Paul Wintz. Digital Image Processing (second edition). Addison-Wesley, 1987
(李树青)

tuxiang zengqiang

图像增强 (image enhancement) 增强图像中感兴趣部分的信息, 降低噪声, 使图像更清晰或更易于分析的过程和技术。图像增强的目的有二: 一是处理原始图像使它更适合于人的观察; 二是变换图像以方便人或机器的分析和处理。增强过程不会增加图像的内在信息量, 但却能增大所关心的信息的动态范围, 使之更易于检测。图像增强包括灰度变换 (参见图像点运算)、图像平滑、图像锐化 (参见图像邻域运算)、图像几何变换 (参见图像几何运算)、图像伪彩色增强等。

在无法知道有关引起图像退化的定量信息的情况下, 使用图像增强技术可较为主观地改善图像的质量。例如锐化技术可用于减少图像可能存在的模糊, 而不管是何种原因产生的这种模糊。在物理本质上, 运动模糊和聚焦不好产生的模糊是不同的, 但相同的增强技术可能会改善任何一种模糊图像的质量。正因为增强技术并非是针对某种退化所采取的方法, 所以很难预料哪些特定的技术对于给定的应用是最好的。因此图像增强算法一般是交互式的, 并根据具体的应用来选择。尽管如此, 图像增强仍是图像处理中一个非常重要的内容, 因为它可以用于几乎所有的图像处理的应用中。

图像增强技术也经常用于改善图像中某种特征的可探测性, 使人 (或计算机) 更容易观察到。减少图像模糊虽然更易于观察某些感兴趣的特征, 但有时可能需要彻底改变图像的视觉效果以突出重要特征的可观察性, 在这种情况下, 可以把增强理解为增强感兴趣特征的可检测性, 而不是改善图像的清晰度。例如, 图像的边缘增强 (锐化) 处理也可以认为是图像增强, 但其结果图像只突出了景物的轮廓、边界, 而原图中一些平缓变化的细节都被忽视了。

伪彩色增强技术可将灰度图像的不同特征映射为不同的彩色, 灰度图像映射成彩色图像。又例如在广告中所用的图像颜色变换、二值化和轮廓效果, 就是一反以往图像质量的逼真性要求, 以得到一种特殊的艺术效果。

图像处理的基本运算中介绍的大部分技术都可以作为图像增强的基本技术。

参考文献

- 赵荣椿等. 数字图像处理导论 (第二版). 西安: 西北工业大学出版社, 1995
(朱志刚)

tuxing bianhuan

图形变换 (graphics transformation) 将计算机生成的图形进行变换的过程和技术,是计算机图形学中较为基础的内容之一。通过变换可以从简单图形得到复杂图形;可以从某一个图形得到多个其他图形。广义上讲,图形变换包括窗口视区变换、图形几何变换、图形投影变换、观察变换及**图形裁剪**。图形变换的核心问题是坐标的转换。其中,窗口视区变换是用户空间与设备空间坐标之间的转换;图形几何变换是由图形的一个几何位置或形状到另一几何位置或形状的转换;由三维空间到二维平面坐标的转换是图形投影变换的内容;观察变换则是由一个空间到另一个空间的坐标转换问题;图形裁剪是计算出感兴趣的特定范围内的图形坐标问题。

参考文献

1. 唐泽圣,周嘉玉,李新友. 计算机图形学基础. 北京:清华大学出版社,1995
2. 唐荣锡,汪嘉业,彭群生等. 计算机图形学教程(修订版). 北京:科学出版社,2000 (周嘉玉)

tuxing caijian

图形裁剪 (graphics clipping) 将落在窗口或观察空间内的图形映射到视图区中,而将落在外面的图形裁剪掉的过程和技术。窗口一般是矩形或长方体,用窗口对平面图作裁剪称为二维裁剪,这是图形裁剪的基础。如图1所示,设某待裁剪线段的起始点和终点坐标分别为 (A, B) 和 (C, D) ,窗口的4条边框为 x_L, x_R, y_B, y_T ,则线段裁剪后的起始点和终点坐标分别为 (A', B) 和 (C', D) 。

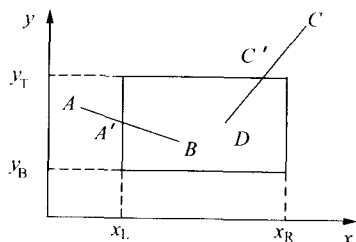


图1 图形裁剪

裁剪包含两部分内容:①点在区域内、外的判断;②计算图形元素与区域边界的交点。由于交点计算比较费时,对于比较复杂的曲线,为避免不必要的计算,可先作曲线的外接矩形与区域的重叠测试。

裁剪算法效率的高低直接影响整个图形系统的效率,要根据实际情况选择不同的裁剪方法。裁剪包括二维线段裁剪、多边形裁剪和字符裁剪等。

参考文献

1. 唐泽圣,周嘉玉,李新友. 计算机图形学基础. 北京:清华大学出版社,1995
2. 唐荣锡,汪嘉业,彭群生等. 计算机图形学教程(修订版). 北京:科学出版社,2000

(周嘉玉 金小刚)

tuxing fanzouyang jishu

图形反走样技术 (anti-aliasing technique)

消除在计算机图形生成中因离散化而引起的误差的技术。又称反混淆技术。计算机图形是由离散点组成的数字化图像,因而每幅图像都是对连续色彩真实画面的一个采样。因此,计算机图形必然与真实景物之间存在误差。这种误差表现为图形上的直线或曲线呈现锯齿状、色彩组成及纹理引起失真,细小物体在画面上得不到反映等。这些问题在**计算机图形学**中称为图形走样或混淆。图形走样在直线和多边形生成、图形线、面消隐、阴影生成(参见**光线跟踪技术**)、纹理生成、光线跟踪和辐射度绘制(参见**辐射度技术**)等情况下都可能以各种不同的方式存在。

从信号理论的角度来看,在图形生成中,由于使用采样技术,当采样频率太低时便会导致图形走样。解决图形走样问题的最根本方法是提高采样频率或者以面积采样代替点采样。提高采样频率的常用办法是提高分辨率,即高分辨率计算,低分辨率(像素分辨率)显示。其实现方法是将要显示的像素划分成许多子像素,然后按通常算法计算出各个子像素的颜色或灰度值,最后将所有子像素的颜色平均值作为像素的显示值。该方法算法简单,但所需要的存储和计算量较大。为了减少计算量,常常使用不均匀地增加采样频率的方法,例如只在图形细节较强或较复杂的部分增加采样频率。采样点的分布也可以是不规则的,这在绘制许多复杂场景时尤其有效。特别是在利用光线跟踪技术绘制,诸如毛绒表面、分数维几何面、几何纹理面等复杂场景时,采用随机分布(如 Monte Carlo 分布)的采样对于反混淆至关重要。事实上,辐射度技术中采用的层次辐射度技术对于几何面片划分的非均匀采样亦起到了反走样的效果。

用面采样代替点采样对于反走样是行之有效的

的。通常在图形绘制时,为了使算法简单都把像素作为一个点来处理。实际上,如果将它看成是一块极小的屏幕区域,将大大减少图形走样现象。此时像素区域与各种图元如直线、多边形相交时,将严格计算相交的面积,其像素的最终颜色值将由与之相交的各种图元的颜色值,利用其相交面积加权平均获得。该方法可成功地用于直线和多边形生成、消隐等的图形反走样。

参考文献

1. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2001
2. Foley J D, Dam A V 著. 交互式计算机图形学基础. 唐泽圣,周嘉玉等译. 北京:清华大学出版社,1986 (吴恩华)

tuxing yuanwenjian biao zhun

图形元文件标准(Computer Graphics Metafile Standard, CGM)

ISO/IEC JTC1/SC24 制定的一种用于图形数据交换的国际标准(ISO/IEC 8632)。它定义了二维图形(图片)信息的计算机可解释的表示,这种表示与任何特定设备或应用无关。CGM 的目的是便于图形数据的储存、检索以及在不同系统、不同应用和不同设备之间的交换。

CGM 标准由两部分组成。其一是定义了一组功能说明,以抽象的词法和句法对图形元文件中的图形元素进行描述;其二是规定了这些抽象元素的3种具体的编码方法。

一个 CGM 文件的头尾是一对方括号(BM)和(EM),文件中可以包含有 n 个图片(n 可以是0或任意正整数)。每幅图片由(BP)、图片描述、图片实体及(EP)组成;图片实体又由(BPB)、控制元素、图元及属性组成。在图片1之前,还有元文件版本号与元文件描述信息。CGM 图形元文件的结构如图1所示。

元文件描述用来说明虚拟设备空间的类型、整

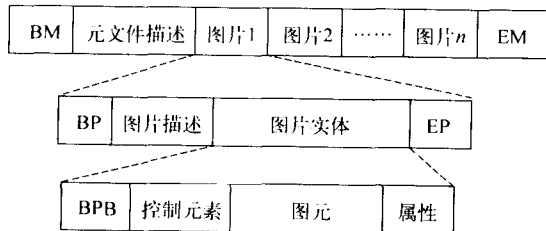


图1 CGM 图形元文件的结构

数及实数的精度、颜色精度、颜色取值范围、字符集表、字体表等。控制元素包括裁剪矩形大小及裁剪标志等。图元有折线、圆弧、椭圆弧、多边形、矩形、椭圆、NURBS、光栅图像、标记、正文等。属性则与图元有关,例如折线和圆弧的属性有线型、线宽、颜色等。

为了满足图形数据的储存和传输的要求,CGM 元文件有3种编码方式。

(1) 字符编码方式 元文件的每个元素均用一个操作码和相关的操作数来表示,且只使用可打印的ASCII字符。这种表示方法比较紧凑,能确保在网络上正确进行传输。

(2) 二进制编码方式 元文件中的每个元素直接使用机内的二进制码来表示,这种做法既有利于文件的生成,也便于以后的解释,其存储开销比字符编码方式略大,不便在网上传输。

(3) 清晰正文编码方式 这种方式以人可阅读的正文方式来表示元文件中的各种元素,它特别有助于对元文件进行编辑和排错操作。

图形元文件 CGM 适合于储存和交换二维图形,在交通、航运、国防等领域有重要的应用。

CGM 标准自1987年发布第1版以来,经过了数次扩充,目前最新的第4版是1999年发布的。近些年来,CGM Open 联盟与W3C合作开发了CGM的一种框架——WebCGM,其主要目的是使CGM图形元文件适合在万维网环境下应用。WebCGM支持如下新的功能:超链接与文档导航、图形的结构化与分层功能、对图形内容的搜索与查询等,它为Web提供了一种发布技术图档、地理数据及其他2D图形的可靠方法。

参考文献

1. ISO/IEC 8632:1999, Information Technology—Computer Graphics—Metafile for the Storage and Transfer of Picture Description Information—Part 1 ~ Part 4
2. WebCGM 1.0 Second Release, <http://www.w3.org/TR/2001/REC-WebCGM-20011217>

(张福炎)

tuyuan shengcheng

图元生成(graphics primitive generation)

根据几何图元参数在图形输出设备上生成图元的过程和技术。通常把构成图形的基本元素称为图元,一般包括点、直线、曲线、圆、多边形、位图和字符等。研究这些图元的生成算法是计算机图形学的基本内

容之一。

在图形输出设备上输出一个点时,需要把该点的坐标信息转换成控制输出设备的相应指令。以阴极射线管监视器为例,需要在指定的屏幕位置上开启(接通)电子束使该位置上的荧光点发亮,因此应将帧缓存中相应坐标位置的内容置为“1”。当电子束“扫视”每一条水平扫描线时,一旦遇到帧缓存中值为“1”的点,就发射一亮光,即输出一个点。

在图形设备上输出一条直线时,需在应用程序中给出该直线端点的坐标,然后由输出设备将对端点间的路径进行绘制。由于大多数图形设备都只提供 x 方向和 y 方向动作的信号,所以只能准确地画出水平或垂直直线。而对于任意斜率的直线,就必须由算法来决定图形设备何时以及应往何方向动作。衡量直线生成算法的优劣,一是看生成的直线是否很好地逼近所需的直线,二是看生成直线的速度如何。由于光栅扫描设备各像素点的坐标是用整数表示的,因此,对直线各轨迹点坐标值的实型数,需作四舍五入,近似成整数值,致使所生成的直线出现台阶或锯齿状。为改进直线的质量,可以采用高分辨率的显示设备,也可采用反走样技术。

利用点和直线的生成算法,也可以生成曲线、圆、多边形、字符等图元。但是,由于这些图元使用频繁,各有特点,通常仍分别研究其快速生成算法。大部分图形加速卡支持用硬件实现点、直线、曲线、圆、位图和字符的绘制,用户只需调用 OpenGL 和 DirectX 中相应的函数即可。

参考文献

1. 唐泽圣,周嘉玉,李新友. 计算机图形学基础. 北京:清华大学出版社,1995
2. 唐荣锡,汪嘉业,彭群生等. 计算机图形学教程(修订版). 北京:科学出版社,2000

(周嘉玉 金小刚)

tuiliji

推理机(inference engine) 计算机系统中实现基于知识推理的部件。基于知识推理是指依据一定的规则,运用知识从已存在的事实导出结论的过程。推理机运用知识库中的知识,按一定的控制策略求解问题。

应用于不同领域的专家系统具有各自的特点,作为其重要组成部件的推理机也要相应地采用不同的推理机制。传统的形式化推理技术是以经典逻辑

为基础的,但随着人工智能研究领域的不断拓展,经典逻辑已不能处理所有问题,各种非经典逻辑推理机制的研究已成为专家系统和人工智能各个领域的重要研究内容之一。①**演绎推理** 属经典逻辑范畴,系指根据公理系统把包含在已知事实中的结论(事实)推导出来。因此,演绎推理的过程并没有增殖新知识。一个演绎推理的逻辑系统是一个无矛盾的公理系统,它根据实际问题新加入的事实,能推出新的结论,这些结论同已有的知识和结论不发生矛盾。演绎推理具有单调性和精确性。②**非单调推理** 是指在逻辑系统中加入新前提后,未必一定导出更多的新结论,反而可能修改或否定原有的结论。常识推理大量地依赖于默认信息,默认信息是指:当且仅当没有事实证明 S 不成立时, S 总是成立的。这种基于默认信息的推理称为默认推理。除了默认推理,还有**约束推理**,可以表示为“当且仅当没有事实证明 S 在更大范围内成立时, S 只在指定的范围内成立”。③**不精确推理** 专家的知识 and 求解问题所用的数据往往是不精确的(或者说具有不确定性和不可靠性),为使专家系统达到高性能,必须解决不确定性问题。处理不确定性有两类方法:第一类是对经典演绎推理的扩展,将具有不确定性的数据和知识与不确定性度量标准对应起来,再给出更新结论不确定性值的算法,从而构成不精确推理模型。第二类是在控制策略一级处理不确定性,通过控制策略的设计来限制减小不确定性对系统的影响。**定性推理**主要起源于现实世界中物理系统的研究。量的定性描述是有明确结构的一种非定量的、非精确的表示方法(如物理系统变化趋势的定性描述)。定性推理就是从物理系统的结构描述出发,导出行为描述和功能描述,预测物理系统的行为,并给出因果关系解释的一种非定量推理方法。

知识的选择和运用原则称为**推理控制策略**。一般说来,专家系统中包含大量知识,因此有效选择与运用知识就成了推理机的关键任务。这一任务由控制策略承担。控制策略能否迅速准确地找到与解有关的知识,直接影响着推理的效果与效率。基于典型的推理控制策略的推理控制有正向推理(也称为数据驱动的控制、自底向上的控制),反向推理(也称目标驱动的控制、自顶向下的控制),以及双向推理(也称混合控制)。

正向推理 由原始数据出发向结论方向的推理。其推理过程为:系统根据用户提供的原始信息,在知识库中寻找能与之匹配的规则。若找到,则将该

知识块的结论部分作为中间结果,利用这个中间结果继续与规则库中的知识匹配,直到得出最终结论。

正向推理简单、易实现,但目的性不强,效率低,需要用启发式知识控制中间结论的选取。另外,由于不能反推,系统的解释功能受到了影响。

反向推理 先提出假设,然后由此出发,通过规则库反复寻找支持假设的证据,即所谓目标驱动方工。当所需的证据与用户提供的原始信息匹配时,推理成功,即假设成立。

显然,反向推理在选择目标时容易有盲目性,因此,反向推理比较适合结论单一或直接提出结论要求证实的系统。

双向推理 先根据原始数据通过正向推理帮助提出假设,再用反向推理进一步寻找支持假设的证据,反复这个过程,直到获得满意的结论。

双向推理集中了正向和反向推理的优点,但其控制策略较前两者复杂。

参考文献

1. 石纯一,黄昌宁,王家庶. 人工智能原理. 北京:清华大学出版社,1993
2. 王士同. 人工智能教程. 北京:电子工业出版社,2001
3. Giarratano Joseph, Riley Gray. Expert Systems-Principles and Programming (third edition). PWS Publishing Company, Boston, USA, 1998 (刘大有)

tuoji dayin shebei

脱机打印设备 (off-line printer) 不需要主机管理,能独立进行打印的**脱机设备**。由于打印机工作速度有限,当需要输出大量数据时,联机打印必然要影响主机的工作效率。因此,大型计算机的大量计算处理结果常先存储在磁带、软磁盘或光盘上,然后通过脱机打印设备打印输出。

脱机打印设备由磁带驱动器(或软磁盘驱动器)、光盘机、打印机以及以微处理器为核心的控制电路组成。控制电路对磁带驱动器、软磁盘驱动器、光盘驱动器和打印机进行状态检测、发出命令等管理工作,把媒体上存储的数据进行处理,通过打印机打印出结果。

(林兼)

tuoji shebei

脱机设备 (off-line equipment) 不受主机控制,也不与其通信的外围设备。它独立进行一些信

息处理工作。因为一般输入输出设备如键盘等工作速度较慢,为了避免影响主机的速度,当输入的数据量较大时,可利用脱机设备先把数据录在磁带、光盘或软磁盘上,再从软磁盘、光盘或磁带输入到计算机中去。计算的结果也可先输出到软磁盘或磁带上,再由脱机设备打印出来。

常见的脱机设备有**数据准备设备**(如数据录入设备)和**脱机打印设备**。数据录入设备已得到广泛应用,用它可组织众多的操作人员进行巨大数据量的录入。对于数据收集工作,如人口普查等,数据录入设备是必不可少的设备。

(林兼)

tuoji shouxie hanzi shibie

脱机手写汉字识别 (off-line handwritten Chinese characters recognition) 通过扫描仪将稿纸上手写的汉字输入计算机并进行汉字识别的过程和技术。

由于不同书写者书写汉字的巨大差异和变化,再加上无法直接获得书写的笔画信息,使脱机手写汉字识别成为最困难的汉字识别问题。手写汉字与标准楷书的差异和变化程度,将直接影响到识别的难度。据此,将手写汉字识别粗略分为规则书写汉字识别与自由书写汉字识别。规则书写汉字是指书写的汉字基本保持横平竖直笔画的手写汉字,而自由书写汉字则包括自由连笔、笔画弯曲的草书书写汉字。

由于手写汉字的巨大不确定性,人们对于识别脱机汉字的方法在相当时间内采取结构分析的方法,即从汉字图像中抽取汉字笔画,按照笔画的属性和逻辑关系构造汉字的结构属性描述,进行结构匹配识别。但是实践证明,这种方法一直未能取得成效。

在印刷汉字识别(参见**印刷体汉字识别**)取得成功的启发下,人们自然地将基于汉字全局信息的统计模式识别方法应用于手写汉字识别。因为,手写汉字识别和印刷汉字识别最主要的差别在于,手写汉字的类内变化要大大超过印刷汉字,仅此而已。这一思路的转变,使脱机手写汉字识别的研究取得空前进展。采取从基元的笔画微结构出发(如四方向线素特征等),利用统计的全局信息进行的统计判决识别,提供了脱机手写汉字识别的基础。

为了较好解决脱机手写汉字识别问题,不仅在字符图像的非线性规一化、特征提取、鉴别特征选择优化等方面做进一步的研究和改进,而且在分类器

设计、鉴别学习算法、支持向量机(SVM)分类诸方面都有相当的发展,促进了脱机手写汉字识别问题的进一步解决。

目前,手写规则汉字的识别问题已经得到较好的解决。自由书写手写汉字的识别也在一定程度上得到了解决。前者的识别率已达98%以上,后者的识别率也可达90%以上。

对于自由书写手写汉字识别的研究的困难,与其说是在算法上,还不如说是在样本的收集上。因为,自由书写手写汉字的随意性,造成字形的随意变化,对于人们都难以辨识的汉字,要求计算机不经过学习就能识别是不现实的。应当说,在有足够样本的条件下,解决自由书写手写汉字应当是可以实现的。

(丁晓青)

W

waibu luyou xieyi

外部路由协议 (exterior routing protocol)

在大型网络中自治系统 (AS) 间使用的路由选择协议。大型的复杂网络往往由许多自治系统组成, 每个自治系统内部有自己的公共管理部门和路由策略, 可以采用自己的内部路由协议。不同的自治系统可以采用不同的内部路由协议。若源和目标处在不同的自治系统中, 当报文分组传送到一个自治系统边界时, 就需要按照外部路由协议表选择路由, 以便进一步传递到另一个自治系统。所以, 外部路由协议是相对于自治系统内采用的内部路由协议而言的。自治系统有时也称为域, 故外部路由协议就是域间路由协议。

内部路由协议和外部路由协议最大的不同在于: 内部路由协议主要关心的是如何选择具有最优开销费用的路由; 外部路由协议可能要经过若干个不同自治系统, 而且每个自治系统所在地域的各方面情况, 以及所用的路由算法及资费标准又可能是各异的, 因而它不仅关心如何找到最佳或最经济的路由, 而且还要关心在考虑许多非技术性的策略因素的情况下如何达到目标。要考虑的策略因素包括自治系统所在地域的政治、安全或经济等多方面的情况。比如说, 某个国家的报文不希望在此传送的过程中通过其敌对国家中的自治系统, 某个公司的报文在传送过程中不希望通过某潜在竞争对手的自治系统以防被窃听等。

Internet 是最大的互联网, 由许多自治系统互连构成。最早在 Internet 中采用的外部路由协议是外部网关协议 (EGP), 发表在征求意见文献 RFC 827 中。后来人们发现 EGP 有不少缺点, 于是在 RFC 1105 中又发表了边界网关协议 (BGP) 来取代它。

参考文献

谢希仁. 计算机网络 (第 2 版). 北京: 电子工业出版社, 1999

(高传善)

wai cunchu shebei jiekou

外存储设备接口 (external storage device in-

terface) 计算机主机与外存储器之间的交接部分。和输入输出设备接口一样, 其基本功能如下:

(1) 地址译码 一个计算机系统通常带有许多种外围设备, 通过接口地址对外围设备进行选择。

(2) 信息交换 通过接口, 计算机把数据和控制信息送给外围设备, 外围设备把数据和状态信息送到计算机。即通过接口实现数据的输入和输出以及控制信息的输出和状态信息的输入。

(3) 数据格式的转换 计算机与各种类型的外围设备在编码、数据格式等方面是不相同的。因此, 接口具有数据的分解或组装、串并行格式的转换、不同编码方式之间相互转换等功能。

个人计算机要做到部件的通用性, 要实现可靠的数据交换, 必须对部件进行标准化, 约定好各种类型的外围设备接受的指令形式和状态控制信息以及通信的方式。这种约定就是接口标准或规范。也就是说, 接口除了电缆、电路芯片、接插座这些硬件之外, 还需要管理程序之类的软件。

在计算机中有两种接口, 即系统级接口和设备级接口。早期外存储器多采用设备级接口, 随着技术的进步, 现在多采用系统级接口。在计算机中常用的外存储设备有硬磁盘驱动器、软磁盘驱动器、光盘驱动器、磁带驱动器等, 下面分别介绍它们的接口。

硬磁盘驱动器接口 硬磁盘驱动器是计算机最重要的, 也是发展最快的外存储设备。早期采用设备级接口, 如 SMD, ST 506/412, ESDI 等 (参见硬磁盘驱动器), 现在多采用系统级接口, 如 IDE, EIDE, SCSI 等。

EIDE, SCSI 这一类并行接口有其固有的缺点, 进一步提高数据传输速率有一定的难度。2000 年以来, 各种串行接口相继问世, 如 USB, IEEE 1394, SATA, SAS 以及 iSCSI 等, 在外围设备中应用越来越多 (参见输入输出接口)。

IDE 接口 是 1984 年 Compaq 公司和 Western Digital 公司提出的。它的特点是将控制器与驱动器做成一体。考虑到当时微型机主要采用 AT 总线, IDE 接口中许多信号直接取自 AT 总线, 原封不动地送往硬磁盘机。适配器只需要做简单的总线缓存和

地址转换,因此适配器的结构也非常简单,安装连接简便,价格也较低。由于它采用了 AT 总线信号,也被称为 ATA 接口。为了和后来开发的 SATA 相区别,有时也称之为 PATA 接口。

ST 506/412 接口规定,硬磁盘驱动器每道为 17 个扇区。由于 IDE 接口的硬磁盘机将控制器和驱动器做在一起,每道扇区数可根据存储密度选择,记录方式也可选择,极大地提高了整机容量。而扇区数和记录格式的不同通过控制器自动转换成 AT BIOS 能支持的驱动器类型。IDE 硬磁盘机内部都有数据缓冲寄存器,读写的数通过接口时采用 8 位并行传送,数据传输速率要比 ST 506/412 高得多,采用 PIO 模式,可达到 3.3 MB/s。

IDE 接口理论上能管理的容量可达 137 GB,但实际上受基本输入输出系统(BIOS)的限制,只能达到 528 MB。

IDE 接口硬磁盘机与适配器之间的连接用一根 40 线的电缆,长度限制为 457.2 mm(18 in)。每根电缆可接主从两台硬磁盘机。

EIDE 接口 对 IDE 进行了改进。IDE 接口的最大缺点是其管理的硬磁盘机容量不能超过 528 MB,这显然不能满足硬盘单机容量日益增长的要求。EIDE 与 IDE 接口完全兼容,与适配器也是通过 40 线的扁平电缆相连,插针信号定义也一样。

与 IDE 相比较,EIDE 有以下特点:

(1) 提供三种硬磁盘机地址管理模式,即普通模式、逻辑块模式(LBA)、大模式。可管理的硬磁盘机容量增大到 8.4 GB。2002 年, big drive 寻址方式(48 位 LBA 方式)将可管理容量增至 144 PB ($1\text{PB} = 10^{15}\text{B}$)。

(2) 可挂接四个外部设备。EIDE 接口标准通常提供两个插座,连接两根电缆,每根电缆可以连接两个硬磁盘机或者其他的外部设备。

(3) 支持多种外部设备。EIDE 支持符合 ATA-PI 接口标准的磁带机和光盘驱动器。

(4) 在高数据传输速率(66 MB/s 以上)时采用 80 线电缆。

(5) 具有更高的数据传输速率。(2002 年) Ultra DMA133 达到 133 MB/s。

(6) 提供两种数据传输模式,即 PIO 模式与 DMA 模式。前者用中断方式由 CPU 管理数据传送;后者用 DMA 通道直接在硬磁盘驱动器与内存存储器之间进行数据传送,减少占用 CPU 的时间。

EIDE 接口是硬磁盘机常用的接口之一。EIDE 接口有许多规范,如: Fast ATA, Ultra ATA, Ultra

DMA 等。它们之间的差别在于有不同的数据传输模式和数据传输速率。

SCSI 接口 是处于主机适配器与智能控制器之间的一种系统级并行接口。SCSI 总线通过适配器与主机相连,通过各种智能控制器和外围设备相连。主要特点如下:

(1) 作为一种通用接口,可连接各种采用 SCSI 接口的外围设备,如硬磁盘机、软磁盘机、磁带机、打印机、光盘机、扫描仪、网络设备等。

(2) 系统的配置灵活。在单一总线上一台外围设备能与多台主机进行通信,一台主机也能与多台外围设备进行通信。构成单主机多控制器系统时,总线上可连接 8 个设备。设备可以是 SCSI 接口的外围设备,也可以是控制器,但其中有一台必须是主适配器。如果有设备是控制器,每台控制器还可以带 7 台设备。利用 SCSI 总线还可以组成多主机多控制器系统,这时总线上的设备数仍是 8 个。总线上的设备分为启动设备和目标设备,启动设备是发出命令的设备,目标设备是接受并执行命令的设备。每一种设备可以是启动设备,也可以是目标设备,总线上的所有设备都能相互通信。

(3) 对硬磁盘机地址的管理,采用连续的数字来编排,也就是只有扇区号。对于硬磁盘机具体的物理参数,有几个柱面、几个磁头或每道有几个扇区,在编程时都不必考虑。

SCSI-1 接口采用 8 根数据线和 1 根奇偶检验线并行传送数据,还有 9 根控制和状态信号线。电缆采用差分驱动和单端驱动两种方式。单端驱动采用一个信号一根导线,所有信号的地线共用一根导线。差分驱动则每个信号都有自己的返回线,要用两根导线(通常采用双绞线)。上述电缆称为 A 电缆,标准规定采用 50 针接插座。SCSI-2 采用 16 位和 32 位并行,定义了采用 68 线的 B 电缆。

SCSI 接口已广泛应用于各种高档微机、工作站、服务器。硬磁盘机、只读光盘机、磁带机和扫描仪等外围设备都有 SCSI 接口的产品。

1986 年美国国家标准协会公布了 SCSI 标准,即通称的 SCSI-1 标准。后来,SCSI 接口也有许多规范,如 SCSI-2, SCSI-3, Fast SCSI, Wide SCSI, Ultra SCSI, Wide Ultra SCSI 等。这些规范有不同的总线宽度和数据传输速率。2002 年发表的 Ultra 320 SCSI 采用 16 位并行传送,数据传输速率为 320 MB/s。

SATA 接口 2001 年秋提出的接口规范,目的是克服 ATA(EIDE)的缺点。它将数据并行传送改为

串行传送,使用4列信号线,即电源、地线、发送数据线和接收数据线,与ATA相比要简单多了。SATA接口对数据和命令都有纠错能力,数据传输电压用250 mV。采用点对点连接,每个通道只接一个设备,数据传输速率可达到1.5 Gb/s。

SAS 接口 2002年4月提出的接口规范其目的也是为了克服SCSI的缺点。SAS接口采用串行传送,采用点对点连接。一条SAS线缆内有两对数据线,分别用于发送和接收。线缆最大长度为10 m。连接对象也不限于启动设备(主适配器)和目标设备(外围设备),还可以是扩展设备。整个系统可连接的设备达到64个,数据传输速率达到3.0 Gb/s。

iSCSI 接口 将SCSI接口技术与以太网技术结合,使存储设备中的数据可直接在以太网上传输的接口规范。

存储网络是个大的复杂的而且昂贵的网络。使用SCSI,成本虽低,但传输数据时会受距离及路径的限制。若使用光纤接口,传输速率高,但结构复杂,投资大。用户需要结构简单、成本低、能够充分利用已有以太网环境完成数据存储工作的技术。因此,iSCSI技术应运而生。2003年2月IETF通过了iSCSI协议,并发布了RFC草案。

iSCSI可以实现在IP网络上运行SCSI协议,支持在系统之间传送标准的SCSI命令。在系统之间的连接是通过标准的以太网的IP网络基础设施实现的,而不是通过SCSI线缆或光纤通道。

iSCSI定义了可靠传输和使用IP路由的TCP中的SCSI信息包的封装。iSCSI协议使得现有的IP网络(LAN、WAN或Internet)能传送整块的数据,而不用修改网络基础设施、主机软件或操作系统,也不用修改目标存储设备。

iSCSI接口在存储区域网(SAN)上已得到应用。

软磁盘驱动器接口 软磁盘控制器通常制成一块集成电路,直接装在主机板上。主机上的插座通过一条34芯扁平电缆与主机箱内部的驱动器相连,可连接两台驱动器。驱动器的插座用34针双列插座,其中奇数全接地。数据是串行传送,只占一位,其余是命令和状态信号。软磁盘接口的数据传输速率很低,有250 Kb/s及500 Kb/s两种(参见**软磁盘驱动器**)。

光盘驱动器接口 常用的有SCSI接口、ATAPI接口、EIDE接口、USB接口、PCMCIA接口等。

ATAPI是ATA的扩充。数据在ATAPI定义的

“包命令”控制下,通过数据寄存器发送。数据传送可以通过DMA方式或PIO方式进行。

磁带驱动器接口 常用的有SCSI接口、ATAPI接口、EIDE接口、USB接口、PCMCIA接口等。

(林兼)

wai cunchu zixitong

外存储子系统(external storage subsystems)

由大容量存储设备和控制设备组成的,用以组织和管理数据的存取和传输的辅助存储系统。

计算机**主存储器**在速度上虽已接近与**中央处理器**相匹配,但因容量有限,价格昂贵,难以实现只有一个层次的存储系统。大容量存储器(如磁带机、磁盘机、光盘机等)由于在容量和成本上的优势,虽存取速度稍慢,但容量巨大。它作为不直接参与中央处理器运算的外存储器,与主存储器相配合,组成了多级存储系统。随着对容量的进一步需求,又出现了容量更为巨大的各种外存储子系统,如自动磁带库、自动光盘库和光盘塔等。这些联机辅助存储子系统处于多级存储系统的最底层,一般把使用频率较低的大量数据存入其中。后来发展起来的**磁盘阵列**,作为一种联机的外存储子系统受到重视。它采用冗余技术来提高可靠性,采用并行处理技术来提高存取速度。磁盘阵列技术对解决主机与输入输出之间的瓶颈问题起了很好的作用。

随着计算机网络的发展,出现了**附网存储(NAS)**、**存储区域网(SAN)**和**光盘镜像服务器**等子系统,使存储子系统更加多样化。

早期的存储系统因单一用户而多采用由主机发出定时脉冲直接控制存储器的存取。后来在子系统中增设了控制器。控制器的功能也由程序控制方式发展到中断控制方式和直接存储器存取(DMA)控制方式,使辅助存储器的操作与主机操作具有初步的并行性。继而在控制器中设置了小型缓冲存储器,使子系统增加了独立完成指令操作的能力。当进一步扩大到具有根据主机程序提供的参数自行编制和执行程序(通道程序)的能力时,数据的存取和状态信息的回收便更具有独立性和并行性,即发展到通道控制方式。通道控制方式仍需主机干预,采用外围处理机方式可使主机的干预减到最少。

外存储器子系统有以下类型。按其配备的大容量存储器的类型划分,有磁带存储系统、磁盘存储系统和光盘存储系统。按数据存取的并行程度划分,

有单用户单通路的存储系统、多通路存储系统以及多机共享的群集存储器系统。按应用的场合划分,有用于科学计算的存储系统,有用于事务处理的存储系统和用于图像处理的存储系统等。通常采用按配备的大容量存储器的类型分类的方法。

高速计算机的外存储器子系统具有多个通路、

多个适配器和控制器,如图1所示。这种系统的控制器使用菊花链式电缆,连接8台磁盘存储器,通过两个主适配器A、B与主机连接。若单台磁盘驱动器的容量为80 GB,则每个通路可提供640 GB的容量,数据传输速率可达80 MB/s以上,且具有高度的并行性与独立性。

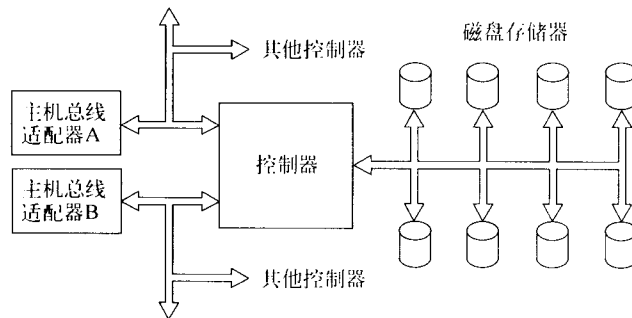


图1 高速计算机外存储子系统

从图可见,外存储子系统硬件由适配器、控制器和大容量存储器三个基本部分组成。适配器处于主机总线与控制器之间,它传送各种命令和数据。控制器则和大容量存储设备相连接,直接控制管理设备的工作,如执行主机的命令,对设备进行寻道、读、写、初始化等操作;对数据进行编码、译码、串并行转换;对设备状态进行检测以及故障诊断,并将设备状态送给主机。所有这些操作,需要软件来支持。

为了兼容,规定了接口标准。介于主机与子系统之间的接口为系统接口,如EISA、ISA、PCI、VME等。介于控制器与辅助存储器之间的接口为设备接口,如ST 506/412、IDE、EIDE、FC、SCSI、SATA、P1394等。接口标准规定了插头插座的物理配置(针数、各针信号安排等)和数据传送规范(串并行、数据传输速率等),使不同的计算机可以采用相同的存储设备。一些接口还允许同一接口连接不同的设备,如SCSI接口可连接磁盘驱动器、磁带机、光盘驱动器,还可以连接打印机等输出设备,有很好的适应性。

随着大规模集成电路的进展,以及大容量存储器的小型化和接口标准的扩展与更新,外存储子系统中各个部分的功能与结构也在变化。

衡量外存储子系统性能的主要指标如下:

(1) 子系统带宽 说明子系统被连续访问时单位时间内能完成的操作数据量。一般采用平均的数据传输速率作为实际的子系统带宽。

(2) 子系统吞吐量 表示子系统在单位时间内能完成的请求数。若外存储器的一次服务所需的平均时间为 T_s ,则吞吐量可表示为 $1/T_s$ 。

(3) 请求的等待时间 指项目请求在排队等待服务时所花费的时间。

(4) 主机利用率 定义为一个进程处理一批数据的主机工作时间与包括数据存取、传输和主机工作时间的总的时间的比值。通常,若主机利用率低,则表示外存储器处于忙碌状态,而主机则相对空闲。

(5) 子系统存储容量 计算机配置的存储容量因用途不同而相差悬殊。对于科学与工程计算、事务处理、气象预报、能源勘探、人口普查等应用都要求较大的容量。某些数据处理系统配备的外存与主存容量之比超过1000:1。

不同的应用领域对系统性能有不同的要求。用于科学与工程计算的超级计算机,它的外存储子系统以顺序输入和输出为主要特征,即数据周期地、大块地从主存到外存往返传输。因此它要求子系统有较高的带宽和很少的请求等待时间,而对吞吐量则要求甚低。用于事务处理的计算机,其子系统以大量地、频繁地随机存取为特征,每次传输的数据量很少,故要求子系统吞吐量大,请求等待时间短,而带宽不要求很宽。用于图像处理的计算机,其子系统以大宗数据往返传输为特征。由于处理比较规范,操作比较单纯,故要求带宽较宽,等待时间较少,吞

吐量则不必很大。为获得最好的外存储器子系统性能,无论在系统组成、设备选择或控制程序设计上都要针对应用场合给予充分考虑。

数字信息存储是计算机、通信等领域不可缺少的重要组成部分。随着因特网与新一代计算机的出现以及图书资料大规模数字化的实现,对存取速度和存储容量的要求日益高涨,它将促进新的外存储器系统的发展。

参考文献

1. 张江陵,冯丹. 海量信息存储. 北京:科学出版社,2003
2. Katz R H, et al. Disk Systems Architectures for High Performance Computing. Proceedings of the IEEE, 1989,77(12) (张江陵)

wailianwang

外联网(Extranet) 在内联网基础上,通过企业间的计算技术形成扩展的企业,使企业延伸到客户、供应商、合作伙伴,甚至包括竞争对手,从一个有形的企业变成一个虚拟企业的网络。这种变化的结果,不仅可使企业降低成本、加速通信和提供及时的信息,更重要的是改变了现存的人际交往和组织间通信方式,建立起新的人际关系和组织间的关系,改变了企业和外界的关系。

参考文献

1. 胡道元. INTRANET 网络技术及应用. 北京:清华大学出版社,1998
2. Douglas E Comer. Internetworking with TCP/IP, Vol. 1, 3rd ed. Prentice Hall Inc., 1995 (胡道元)

wanquan pianxu

完全偏序 (complete partial order, CPO)

具有某种完备性质的偏序集。它在 λ -演算和指称语义方面有着广泛的应用。

设 D 为集合, \leq 为 D 上的二元关系。称 $\langle D, \leq \rangle$ 为偏序集,如果它满足:①对每个 $x \in D$ 皆有 $x \leq x$ (自反性);②如果 $x \leq y$ 且 $y \leq x$, 则 $x = y$ (反对称性);③如果 $x \leq y$ 且 $y \leq z$, 则 $x \leq z$ (传递性)。为方便起见,常把偏序集 $\langle D, \leq \rangle$ 简称为 D 。如果偏序集 D 有最小(或大)元,则用 \perp (或 \top) 表示。并称为 D 的底(或顶)。

具有最小元 \perp 的偏序集 D 称为平坦的,是指对

任意的 $x, y \in D$ 皆有 $x \leq y$ 当且仅当 $x = \perp$ 或 $x = y$ 。

设 S 为偏序集 D 的一个非空子集。 D 中的元素 x 称为 S 的上界,如果对每个 $s \in S$ 皆有 $s \leq x$ 。如果对 S 的每个上界 y 还都有 $x \leq y$, 则称 x 为 S 的上确界。如果 S 的上确界存在,则它必是惟一的。

设 S 为偏序集 D 的非空子集。如果对任意的 $x, y \in S$, 都必有 $z \in S$ 使 $x \leq z$ 且 $y \leq z$, 则称 S 为定向的。如果偏序集 D 满足:(1) D 有最小元,(2) D 的每个定向子集都在 D 中有上确界,就称 D 为一个完全偏序。

平坦偏序集总是完全偏序。其他一些完全偏序的例子有:具有最小元的有限偏序集; ω 序数连同定义在其上的小于等于关系“ \leq ”;集合 S 的幂集连同定义在其上的包含关系“ \subseteq ”;实单位闭区间 $[0, 1]$ 连同定义在其上的小于等于关系“ \leq ”。

设 $\langle D_1, \leq_1 \rangle$ 和 $\langle D_2, \leq_2 \rangle$ 都是完全偏序。若在 $D_1 \times D_2$ 上定义二元关系“ \leq ”如下:若 $x_1, x_2 \in D_1$ 且 $y_1, y_2 \in D_2$, 则 $\langle x_1, y_1 \rangle \leq \langle x_2, y_2 \rangle$ 当且仅当 $x_1 \leq_1 y_1$ 且 $x_2 \leq_2 y_2$, 则 $\langle D_1 \times D_2, \leq \rangle$ 是一个完全偏序,称为 $\langle D_1, \leq_1 \rangle$ 与 $\langle D_2, \leq_2 \rangle$ 的积。

设 $\langle D, \leq \rangle$ 为一个完全偏序。如果 $O \subseteq D$ 满足:

(1) 若 $x \in O, y \in D$ 且 $x \leq y$, 则 $y \in O$;

(2) 若 X 为 D 的定向子集且 $\cup X \in O$, 则 $X \cap O \neq \emptyset$ 。

则称 O 为 D 的 Scott 开集。若令

$$\mathcal{C}_D = \{O \mid O \subseteq D \text{ 且 } O \text{ 为 } D \text{ 的 Scott 开集}\},$$

则 $\langle D, \mathcal{C}_D \rangle$ 就构成一个拓扑空间,并称为 \mathcal{C}_D 为 D 上的 Scott 拓扑。这个拓扑空间是 T_0 空间,但不一定是 T_1 空间。

(王兵山 王水汀)

wanweiwang

万维网(world wide web, WWW) 基于超媒体的,方便用户在 Internet 上检索和浏览的一种广域信息查询工具。

超媒体是超文本和多媒体在信息浏览环境下的结合,用户使用万维网(WWW)不仅可以查询文本信息,还可以获得声音和图像信息。超文本是以网状方式链接的电子文本。将文档中不同的部分通过关键字的方式链接起来,使得信息不仅可以用传统的线性方式,还可以用交互方式搜索。这是一种全局性的信息结构,在一个文档中,只要选中关键字,就可以进入与关键字链接着的另外一个文档,这个文档可以是在同一台机器上,也可能是在 Internet 的

其他服务器上,在这个文档中可含有多个超文本链接。

万维网基于客户-服务器模式。万维网客户机为用户提供基于超文本传送协议(HTTP)的用户界面。万维网服务器的信息是用超文本置标语言(HTML)来描述的。HTML文档由文本、格式代码和到其他文档的链接所组成。其中超媒体链接使用统一资源定位器(URL)。URL是标准的寻址机制,用来定位检索在万维网上任何地方的文档。它由3部分组成:①所使用的传送协议;②服务器地址;③定位在该服务器上的文件的全路径名。如果第一部分为http://,则为超文本传送协议,支持超媒体万维网环境。URL的第一部分也可以是gopher,ftp,telnet,wais,news等协议。这样,用户可通过万维网方便地访问因特网上的其他信息资源。

由于万维网是超文本和超媒体信息服务,它的应用发展十分迅速。在因特网上的万维网服务器已超过几万个。(胡道元)

Wanweiwang shuju guanli

万维网数据管理(Web data management)

万维网环境下各种复杂信息的有效组织与集成、准确查询和发布的技术。

从某个角度讲,万维网(Web)就是一个巨大的、分布的、结构松散的数据库,是信息的集合体。但严格地说,万维网中的信息不是按照某个数据模型组织和管理的,它又不是数据库。万维网数据管理是一个新的研究领域,它融合了网络技术、数据库技术、信息检索技术、数据挖掘技术等多个研究领域的技术。

万维网数据管理的一个核心问题是万维网数据集成。面对内容庞杂、动态变化的万维网信息资源,人们很可能身陷信息的海洋而无所适从。万维网数据集成就是将各种数据源集成起来,从而提供一个统一的接口供用户开发万维网应用。

目前,建立万维网数据集成系统的方法主要有:数据仓库方法和包装器-中介器方法。

在数据仓库方法中,各数据源的数据按照所需的全局模式从各数据源提取并转换,储存在数据仓库中。用户的查询就是对数据仓库中的数据进行查询。该方法的优点是建立系统的过程简单。但是,由于万维网数据源包含海量的数据而且数据是不断变化的,数据仓库的集成系统面临着初始数据装载和变化后数据的更新问题。

包装器-中介器方法并不将各数据源的数据集

中存放,而是通过包装器-中介器体系结构满足上层的集成应用的需求。这种方法的核心是中介模式。数据集成系统通过中介模式将各数据源的数据集成起来,而数据仍储存在局部数据源中,通过各数据源的包装器对数据进行转换使之符合中介模式的要求。包装器-中介器方法解决了数据更新的问题,从而弥补了数据仓库方法的不足。但是,由于各个数据源的包装器是要分别建立的,因此,万维网数据源的包装器自动建立问题又给人们提出了新的挑战。

万维网上的数据越来越多地将以可扩展置标语言(XML)的形式存在,这给万维网数据管理提出了新的研究问题,即XML数据管理。XML数据管理涉及的问题很多,包括XML的数据模型和理论研究,XML数据模式的研究,XML查询语言,XML查询处理和优化,XML数据的高效存储和索引,以及XML数据视图等问题。

基于可扩展置标语言(XML)的查询语言有XQL,XSL,XML-QL,Lorel,Quilt,XQuery等。XQuery(W3C,2001d)是目前有代表性的XML查询语言,它来源于XML查询语言Quilt(Chamberlin等,2000),主要借鉴了XPath,XML-QL,SQL,OQL,Lorel,XQL和YATL等。为解释XML查询语言,人们提出了可扩展置标语言(XML)代数。XML代数一般为嵌套关系代数的扩展。XML代数具备了嵌套关系代数的主要特征,并出现了一些新的操作符。它的基本操作是将查询模式体(树结构)与XML文档实例(图结构)进行条件匹配,产生满足模式体结构的结果树,这一过程通常称作模式-实例绑定。XML数据存储方法通常有四种:基于关系数据库、面向对象数据库、直接储存成XML文本文件和专门设计的XML数据存储策略。

参考文献

Abiteboul S, Buneman P, Suciu D. Data on the Web: From Relations to Semistructured Data and XML. California, Morgan Kaufmann, 2000 (孟小峰)

wangge jisuan

网格计算(grid computing) 通过互联网络,将不同空间位置、不同类型的物理与逻辑资源以开放和标准的方式组织起来,通过资源共享和动态协调,来解决不同领域的复杂问题的分布式和并行计算。网格计算改变了人们对计算的传统看法,也改变了人们传统的解决问题的方式。网格计算是借鉴

传统电力网的概念提出来的。在网格计算中,人们能使用一种来自网络的计算能力与资源,无须知道网格资源的确切提供者以及提供者的地理位置,只需集中精力考虑如何使用这些资源与能力来解决更富有挑战性的问题。

发展简史

网格计算的研究可以划分为三个阶段:

第一个阶段是从 20 世纪 80 年代末到 90 年代初期,是网格计算的萌芽阶段。这时出现了比较成熟的千兆网技术、元计算系统和**集群式计算系统**,这些都为网格计算的出现提供了必要的条件。

第二个阶段是从 20 世纪 90 年代中到 90 年代末,是网格计算的早期研究和实验阶段。此时出现了一些实验项目,比如 I-WAY 等,还有不少学术性的研究项目和应用,这些项目对以后的网格计算研究有很大的影响,比如 Globus。

第三个阶段是从 21 世纪初到现在,是网格计算迅速发展和广泛应用的阶段。这时网格计算的概念已经被广为接受,越来越多的国家、地区与组织都在积极开展各自的网格计算项目,一些大型的世界性网格组织如 GGF 和重大网格计算应用已经开始发挥作用。但是还存在不少问题需要解决,突出的问题就是标准化的问题,当大量的网格计算系统和应用出现以后,如何制定一个统一的标准来规范化这一新兴领域是一个亟须解决的问题。开放式网格服务体系结构(OGSA)就是这种努力的结果,有不少组织在开展这方面的工作。

组成与特点

网格系统一般从下到上分为三个主要组成部分(如图 1 所示):①网格基础构件;②网格管理与服务系统;③网格应用。

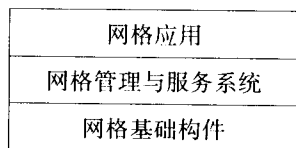


图 1 网格系统层次结构

网格基础构件包括网络和其他的各种网格物理资源结点两大组成部分,网格物理资源由各种不同类型的计算资源、数据资源、信息资源、仪器设备等组成。这些资源所处的地理位置是分布的。网格资源结点之间通过互联网络或其他高速网络连接起来,组成网格向外提供各种资源与服务,实现资源的

共享与协作。同时每个物理资源结点又是一个具有自主功能的相对独立的系统,可以有自己独特的管理和使用策略。网格管理与服务系统通过实现对网格基础构件的统一管理,为各种不同的网格应用提供服务。网格管理包括安全性保证,资源的动态发现、管理与协作,跨地域的任务调度,异构系统的协作,网格协议的管理与实现等。网格服务是各种网格功能的综合体现,各种不同的应用是通过具体的网格服务来使用网格功能的。网格应用是网格计算的最终目的,通过网格应用可以解决复杂的更具有挑战性的问题。

网格计算一般具有如下特点:①灵活性与扩充性,即网格计算的方式不仅十分灵活,而且网格资源是可以不断扩充的,网格计算系统是可扩展的;②局部自治性,网格资源结点是一个相对独立的系统,具有独立自主性;③全局名字空间,即在不同的地方,用户都可以用相同的名字和密码对网格计算系统进行访问,不会因为地理位置和具体系统的不同而有所改变;④透明访问,即网格资源的位置、类型等特性对用户是透明的;⑤高性能与高质量,即网格系统必须能够提供更高的性能,同时对提供的服务保证高质量;⑥安全性,即对网格系统的使用是有安全保障的,不用担心秘密或者私人信息被他人截获或者泄漏;⑦异构性与可移植性,即网格系统可以容纳不同类型的资源,同时保证网格应用可以方便地在不同的网格系统上移植;⑧容错性,即网格系统必须能够对用户或者系统的异常错误能够容忍和恢复,不致造成巨大的损失;⑨动态性,即网格系统应该能够动态适应底层计算资源和上层网格应用的变化。

网格概念的核心与实质

网格概念的核心是资源(包括服务)以及对资源的使用,这里的资源包括计算机、数据库、仪器设备、信息服务等极其广泛的内容。网格概念背后的实质,是在打破传统的强加在资源之上的种种限制的基础上,为使用者提供一种前所未有的高级服务。所谓打破对资源的限制,包括下面几个方面的含义:①资源的网格化,即将资源从特定的地理位置的束缚中解放出来,使得该资源可以通过网格输送到任何角落,达到网格资源与地理位置分离的目的;②网格资源的协调,即在一定的规则约束和管理下,任何网格资源都可以实现相互协作,打破不同资源之间在广泛共享与协作方面的障碍;③网格资源的融合,可打破原来在资源能力和资源类型

方面的限制。因此,网格系统提供的资源是增强和放大后的可以动态组合与使用的资源。

关键问题与技术

网格计算的关键问题包括网格体系结构与网格标准化问题。网格作为一个庞大的信息基础设施,必须为它设计一个坚实的体系结构,网格体系结构是支撑网格有效正常运转的基石,因此必须重点解决网格体系结构所面临的种种问题。

网格计算面临的另外一个问题就是标准化的问题。网格计算走向实用,必须建立一个共同遵守的有效的标准。只有这样,才能够完成已有系统的网格化改造,才能够将新开发出来的系统方便地融入到已有的网格系统之中,避免由于标准不统一造成在效率、性能、共享等方面的问题。

网格计算还存在着一些关键的技术问题有待解决。包括:网格资源的监测与发现技术,随着网格系统包括的资源越来越多,网格资源的准确发现、定位与监测是正确网格调度的前提,是网格资源得以有效使用的前提;网格资源管理,通过对网格资源的动态高效的分配、调度以及管理,可以实现对网格资源的有效、充分使用,并为网格应用提供高质量的支持;传统资源的网格化包装问题,各种传统的物理资源,只有通过特定的形式转化为网格可以使用的资源,才能够充分发挥其作用;各种网格应用门户的开发技术,用户是通过网格门户来使用网格的,因此网格门户的开发与实现技术直接影响到网格能否被广泛接受和使用。

研究类别

关于网格计算的研究,可以从如下几个方面进行划分。

第一个方面是关于网格基础设施的研究和开发项目,其中最具有代表性和影响力的就是 Globus 项目。该技术目前支持的几种典型应用包括:①分布式超级计算;②快捷仪器,通过访问数据文档和提供在线处理能力而增强科学仪器的功能,比如 X 射线 CMT 应用;③桌面超级计算,即通过桌面系统就可以访问到超级计算能力;④远程沉浸,通过将模拟技术、虚拟现实技术和协同工作环境相结合,提供一个共享的虚拟设计空间,这种应用对计算和网络的性能要求很高。

第二个方面是网格系统建设的研究,包括简化用户与网格资源之间的交互,分布的科学团体之间的支持密集计算和大数据量分析的计算基础设施,实现分布式的高性能计算和对科学与工程计算中大

粒度数据的管理等研究。

第三个方面是网格计算应用和相关库开发方面的工作,由于网格的最终目的还是为了支持各种应用,因此面向不同的应用领域开发网格应用和相关的库也是十分重要的工作。

第四个方面是商用网格计算的研究,已经有了一些初步的产品。

几个典型的网格计算试验床

大型的网格计算试验床可以支持大量的用户群,而且能够使得不同学科的科学和工程研究人员在试验床上开展富有成效的研究工作。主要有以下几个研究项目:

(1) 美国国家技术网格 它由 NPACI 和 NCSA 两大组织负责建立该网格计算系统。他们运用 Globus 将各种设施和资源,包括超级计算中心,重点研究实验室,学院和大学的校园连接起来,建立网格计算系统,并且鼓励使用原型网格计算系统,支持分布式的科学与工程计算与应用,形成全美范围的计算基础设施的基础,就像当初的 NSFnet 对后来的互联网所起的作用一样。

(2) 欧洲的数据网格 它由 6 个主要的联合伙伴 (CERN, CNRS, ESRIN, INFN, NIKHEF, PPARC) 和 15 个相关的成员组成,它在网格技术的支持下希望达到如下目的:①建立一个研究性网络,开发相应的技术,以建立一个更大规模的全球数据网格;②让真正的用户介入,通过大规模的应用来验证这种技术的有效性;③展示建造、连接和有效管理大规模通用目的的低成本数据密集型集群式计算机的能力。

(3) NASA 的 IPG 运用 Globus 工具,将加入到其中的组织的超级计算机和存储设备连接起来形成一个单一、无缝的计算环境。在政府、学术界和工业界的共同努力下,IPG 可以帮助美国科学家协同解决 21 世纪面临的重要问题,正如 Web 使得任何地方的信息都可以从所有的地方访问一样,IPG 希望给美国的研究者和工程师,在他们需要的任何时候,提供远距离超级计算资源和数据仓库的能力。

(4) ASCI 的分布资源管理 (DRM) 试验床 美国能源部的 Lawrence Livermore、Los Alamos 以及 Sandia 三个重点实验室在从事 ASCI 的研究,它是美国核计划的重要组成部分,这些实验室在建造 DRM 的试验床,用于管理所有实验室的计算资源。使用的是 Globus 工具系统,支持 Kerberos 安全。

(5) GUSTO 试验床 用来测试 Globus 工具箱。

它由一些联合伙伴共同建造,到2000年2月为止,它已包括了23个国家的125个地点,是最大的计算环境之一。

发展趋势

网格计算正处在急剧膨胀和发展的阶段。虽然网格计算的技术难点还没有完全解决,但是已经跨越了以前的探索与实验性研究的阶段。其研究中心将转移到网格计算标准的制定和开辟真正有效的网格计算应用。

OGSA代表了网格计算体系结构的新发展和标准,它将网格中的一切资源都抽象为服务,借鉴了Web服务的框架,并结合已有网格研究的成果,将传统的科研领域的研究和商业领域的已有成果有机集成,形成一种同时适合两者的体系结构。

网格计算的研究出现了多样化的趋势,从原来的主要以计算为主的研究发展到全方位的网格服务研究。网格系统的应用领域一方面在扩展,另一方面又在细化,因此出现了不少专用的网格系统,如地震网格,商用网格等。

参考文献

1. Ian Foster, Carl Kesselman(eds.). The Grid: Blueprint for a New Computing infrastructure. San Francisco: Morgan Kaufmann Publishers, 1999
2. Foster I. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. In: First IEEE/ACM International Symposium on Cluster Computing and the Grid, 2001. Proceedings. Page(s): 6~7
3. Ian Foster, Carl Kesselman, Jeffrey M. Nick, Steven Tuecke. Grid Services for Distributed System Integration. IEEE Computer, 2002, 35(6): 37~46

(都志辉)

wangguan

网关(gateway) 位于开放系统互连基准(参考)模型(OSI/RM)的高层(运输层到应用层)的一种网络互连设备。它是由软件或者硬件实现的不同网络和网络应用的交接点。它在不同的网络之间翻译地址、转换协议、变换数据格式等,具有很大的灵活性。

从网络的角度,网关可用于互连采用不同协议的两个网络,这两个网络在低层可以是同一个网络。例如,在同一个以太网上,一部分工作站运行TCP/IP网络软件,另一些工作站运行DECnet网络软件,这种情况下的以太网实际上分成了两个逻辑网:

TCP/IP网和DECnet网。当在此低层以太网上使用网关时,就相当于把这两个高层逻辑网互连。更一般的情况是用网关来连接两个具有不同的网络协议且物理上互相独立的网络。这两个网络类型可以是不同的,如局域网和广域网、以太网和令牌环网;也可以是相同的,但具有不同的高层协议。防火墙和代理服务器等也是一种网关。

从网络应用的角度,网关又可分为邮件网关、支付网关、无线应用网关、反病毒网关、内容过滤和安全网关等。例如,支付网关使持多种银行卡的顾客可以通过同一个支付网关在线支付,否则,顾客需要通过各自不同银行卡的支付网关进行支付,这样花费很大而且耗时。又如,反病毒网关具有反病毒保护、监视ActiveX和Java applet、过滤URL和基于关键字的E-mail拦截等功能。它可以对照自带的不可信任地址列表,扫描和拦截可能带有病毒的E-mail和网页。

(张世永)

wangji xieyi

网际协议(internet protocol, IP) Internet采用的一种不可靠的、无连接传送协议。IP提供了三种功能:

(1) 定义了在互联网中数据传送的基本单元,规定了TCP/IP互联网上传送的数据格式(参见Internet体系结构);

(2) 其软件完成路由选择功能,选择数据传送的路径;

(3) 包含了一组分组处理、差错信息发生以及分组丢弃等不可靠分组传送的规则。

TCP/IP互联网的基本传送单元是IP数据报,包括报头部分和数据区部分。在数据报格式中有一项服务类型规定了该数据报的处理方式。数据报通过底层的物理网传输,为使互联网传送更有效,要保证每个数据报都用单独的物理帧传送。

物理网络(参见Internet体系结构)将包括数据报报头的整个数据报作为数据。在理想情况下,将整个数据报封装在一个物理帧中,使物理网络上的传送十分有效。为此,可以选择一个最大数据报长度,使得数据报总能完整地放到一个帧里。但实际上,不同类型的物理网络对一个物理帧可传送的数据量规定了不同的上限,称为网络最大传送单元(MTU)。IP软件必须选择一个合适的初始数据报大小,同时,对MTU较小的网络,提供一种把大的数据分成较小的单位的方法,这种较小的单位被称

为段。

数据报的重新组装有两种方法:一是在分段的数据报通过一个网络后就将其重组;二是在到达目的主机后重组。一般来说,后者较好,它允许对每一个数据报段独立地进行路由选择,且不要求路由器对分段存储或重组。

数据报格式中有一个生存时间字段,用来设置该数据报在互联网中允许存在的时间,以秒为单位。

在一个分组交换系统中,路由选择是指选择一条路径来发送分组的过程。可以把路由选择分成两种方式,即直接传送和间接传送。

在同一物理网上,两台机器之间 IP 数据报的传送不涉及路由器。发送方将数据报封装在物理帧中,将目的 IP 地址(参见 Internet 地址)和一个物理硬件地址绑定在一起,并将产生的帧直接传送到目的站。由于 IP 地址分成网络标识和主机标识两部分,通过判别网络标识就能确定源站和目的站计算机是否在同一物理网上。由于数据报从源站点到目的站的路径上的最后一个路由器与目的站点连在同一物理网上,因此最后一个路由器将使用直接传送方式。

当源站和目的站通过多个路由器互联时,数据报的传送只能采用间接传送方式。发送主机通过连接的网络将数据报传送到一台路由器。一旦帧到达该路由器,封装的数据报就被提取出来,以便根据报头中信息选择通往目的地的路径上的下一台路由器。这个过程直到数据报到达某台可以将其直接传送的路由器为止。

为了进行路由选择,需要路由选择表。该表储存各个目的站点以及如何到达目的站点的信息。

通过路由器转发数据报,IP 软件提供了可靠的无连接数据报传送服务。假如路由器不能正确选择路由或传送数据报,或者它检测到一个异常条件影响它转发数据报,路由器需要通知源站点采取措施避免或纠正出现的问题。为了使 TCP/IP 互联网中的路由器能报告差错或提供有关意外情况的信息,在 TCP/IP 协议中设计了一个特殊用途的报文机制,称 Internet 控制报文协议(ICMP)。它是 IP 的一部分,并在每个 IP 实现中都是必需的。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999
2. Douglas E Comer. Internetworking with TCP/

IP, Volume I, 3rd ed. Prentice Hall Inc., 1995

(胡道元)

wangluo anquan

网络安全(network security) 计算机网络的机密性、完整性和可用性的集合。机密性指通过加密数据防止信息泄露;完整性指通过验证防止信息篡改;可用性指得到授权的实体在需要时可使用网络资源。由此还可派生出一些属性:可审性、真实性、可控性、抗否认性等。可审性指通过基于身份标识和鉴别服务的审计提供过去事件的记录;真实性指通过身份鉴别来确定发送方和接收方的真实身份;可控性指通过访问授权控制用户对网络资源的访问权限;抗否认性指通过数字签名提供有效证据防止事后抵赖。

网络安全包括信息载体的安全和信息自身的安全两个方面。维护信息载体的安全就要抵抗对网络和系统的安全威胁。这些安全威胁包括物理侵犯(如机房侵入、设备偷窃、废物搜寻、电子干扰等)、系统漏洞(如旁路控制、程序缺陷等)、网络入侵(如窃听、截获、堵塞等)、恶意软件(如病毒、蠕虫、特洛伊木马、信息炸弹等)、存储损坏(如老化、破损等)等。为抵抗对网络和系统的安全威胁,通常采取的安全措施包括门控系统、防火墙、防病毒、网络入侵检测、漏洞扫描、存储备份、日志审计、应急响应、灾难恢复等。

维护信息自身的安全就要抵抗对信息的安全威胁。这些安全威胁包括身份假冒、非法访问、信息泄露、数据受损、事后否认等。为抵抗对信息的安全威胁,通常采取的安全措施包括身份鉴别、访问控制、数据加密、数据验证、数字签名、内容过滤、日志审计、应急响应、灾难恢复等。

完整的信息安全保障体系应包括保护、检测、响应、恢复四个方面。保护是指用加解密技术、访问控制技术、数字签名技术对信息的传输、存储、访问加以保护。检测是指对信息传输的内容的可控性检测、对信息网络访问过程的检测、对违规与恶意攻击的检测、对系统与网络的安全漏洞的检测。响应是指保证信息系统与网络遇到攻击时及时采取措施,提供有力的响应机制。恢复是指灾难恢复以及评估系统受到的危害与损失。

国际标准化组织 ISO 在开放系统互连基准(参考)模型中定义了具有七个层次的网络互连模型。相应地,在各个层次需要提供不同的安全机制和安服务。在物理层要保证通信线路的可靠,不易被窃

听。在链路层可以采用加密技术,保证通信的安全。在网络层可以采用传统的防火墙技术和IP加密传输信道技术(IP SEC),在两个网络结点间建立透明的安全加密信道,这适用于在公共通信设施上建立虚拟的专用网。在传输层,可以实现进程到进程的安全通信。如现在流行的安全套接字层(SSL)技术,在两个通信结点间建立安全的传输控制协议(TCP)连接。针对专门的应用,在应用层实施安全机制,对特定的应用是有效的。基于能提供各种安全服务的**安全中间件**技术是一种通用的应用层安全服务。它克服了针对专门应用实施安全服务的缺陷。安全中间件是基于操作系统之上的一层系统软件,它独立于操作系统,便于在不同的平台和操作系统中实现。

网络安全的实现是一个复杂的系统工程,包含5个关键阶段的连续过程:评估、策略、实施、培训和审计。网络安全评估包括网络与信息资产评估、威胁评估、漏洞评估、风险评估以及预防的对策评估等。策略及过程阶段应确定该组织期望的安全水平以及在实施时需做的工作,包括信息策略、安全策略、用户策略、后备策略、账户管理过程、事故处理过程、灾难恢复计划等。实施阶段包括技术和工具的选择、物理控制以及安全人员的聘用。某些安全的实施需改变系统配置,需要介入系统和网络的管理。培训阶段需建立对有关人员(包括员工、管理员和开发人员)提供信息安全教育的机制。审计是网络安全全过程的最后阶段,以确保安全的控制能满足所制定的策略。

参考文献

胡道元,闵京华. 网络安全. 北京:清华大学出版社,2004

(胡道元)

wangluo caozuo xitong

网络操作系统(network operating system, NOS) 能够控制计算机传送信息和共享资源,为网络用户提供各种**网络服务**的相关软件的集合。网络操作系统是网络用户和计算机网络之间的接口,对计算机网络系统的性能有着至关重要的影响。

网络操作系统除了具有通常**操作系统**的各种管理功能外,还必须提供以下几项功能:①网络通信在源和目标计算机之间实现无差错的数据传输;②网络资源管理 对网络中诸如硬盘、打印机、文件和数据等所有硬、软件资源实施有效管理,协调多用户对共享资源的使用;③网络管理 如安全控制、性能监视、网络维护等;④网络服务 如远程作业

录入和打印,电子邮件,文件传输等。

典型的网络操作系统具有以下特征:①硬件独立性 可以运行在不同的网络硬件上,可以通过**网桥或路由器**与别的网络连接;②多用户支持 应能同时支持多个用户对网络的访问,应对信息资源提供安全和保护功能;③其他服务 支持网络实用程序及其管理功能以及支持多种客户端和多种增值服务。

随着**计算机网络**的飞速发展,出现了多种网络操作系统,其中主流产品是:Netware、Windows NT以及UNIX、Linux等四种。Netware在文件共享、打印共享方面有很好的表现,是微机局域网的网络操作系统的理想选择;Windows NT工作在微机和工作站上,是一个界面友善、功能较强和易于管理的网络操作系统;UNIX以其高效、稳定的特点适用于任务重大、高端关键的应用场合,是惟一能跨多种平台的网络操作系统;Linux作为UNIX的一个变种,继承了后者的许多优点,同时又有了很大发展,越来越多地被选作为网络操作系统。

参考文献

张尧学等. 计算机网络与Internet教程. 北京:清华大学出版社,1999

(王晓春 费翔林)

wangluo ceshi

网络测试(network test) 对网络设备、网络系统以及网络对应用的支持进行检测,以展示和证明网络系统能否满足用户在性能、安全性、易用性、可管理性等方面的需求。

网络测试的实施一般包括以下环节:①根据测试目的,确定测试目标;②在对相关网络技术和实现细节透彻掌握的基础上,设计测试方案;③建立网络负载模型;④配置测试环境,包括测试工具的选择及必要的测试工具的研发;⑤采集和整理数据;⑥分析和解释数据;⑦准确、直观、形象地表示测试结果。

网络测试包括网络设备测试、网络系统测试和网络应用测试三个层次。

(1) 网络设备测试 主要包括以下几个方面:功能测试、可靠性和稳定性测试、一致性测试、互操作性测试和性能测试等。

功能测试验证产品是否具有设计的每一项功能。可靠性和稳定性测试往往通过加重负载的办法来分析和评估系统的可靠性和稳定性。网络产品不同于其他产品的最大特点是必须符合标准,不同的

网络产品之间要能互操作。一致性测试验证产品的各项功能是否符合标准。如交换机对 IEEE 802.3, IEEE 802.3z, IEEE 802.1P, IEEE 802.1q, IEEE 802.3x 等的支持。互操作性测试考察一个网络产品是否能在一个由不同厂家的多种网络的产品互连的网络环境中很好地工作。性能测试的主要目标是分析产品在各种不同的配置和负载条件下的容量和对负载的处理能力,如交换机的吞吐量、转发延迟等。

典型的网络设备测试方法有两种:第一种将设备放在一个仿真的网络环境中进行测试;第二种方法是使用专用的网络测试设备对产品进行测试。

(2) 网络系统测试和网络应用测试 网络系统测试除了普通意义上的物理连通性、基本功能和一致性的测试以外,主要包括网络系统的规划验证测试、网络系统的性能测试、网络系统的可靠性与可用性的测试与评估、网络流量的测量和模型化等。

网络系统的规划验证测试主要采用的两个基本手段是模拟和仿真。模拟是通过软件的办法,建立网络系统的模型、模拟实际网络的运行。通过设定各种配置和参数模拟系统的行为,对系统的容量、性能以及对应用的支撑程度给出定量的评价。这对于大型网络的规划设计是不可缺少的环节。仿真是指通过建立典型的试验环境,仿真实际的网络系统。规划验证测试的目的在于分析所采用网络技术的可行性和合理性,网络设计方案的合理性,所选网络设备的功能、性能等是否能够合理地有效地支持网络系统的设计目标。

网络系统的性能测试是通过对网络系统的被动监测和主动测量确定系统中站点的可达性、网络系统的吞吐量、传输速率、带宽利用率、丢包率、服务器和网络设备的响应时间、哪些应用和用户产生最大的网络流量,以及服务质量等。此项工作同时可以发现系统的物理连接和系统配置中的问题,确定网络瓶颈,发现网络问题。

网络应用层次上的测试则主要体现在测试网络对应用的支持水平,如网络应用的性能和服务质量的测试等。

网络系统测试的核心工具是协议分析仪。这是一种专用的网络测试设备,它能够连接到网络上,产生并向网上发送数据、捕捉网上数据、分析数据。协议分析仪一般具有网络监测、故障查找、协议解码和流量产生等功能。

网络流量的测量和模型化对于分析网络性能和

带宽的利用率,指导网络流量管理,开发高效的网络应用十分重要。这方面的工作主要有:①产生已知特征的流量,使该流量沿网络传播,最后回到测试仪。记录和分析流量特性的任何改变(如延迟漂移)。②对链路总体流量的测量和传输时间、吞吐量、带宽利用率等的分析。③分析特定流量的特征和提供的服务质量(QoS);收集一个时间段的测量数据进行分析,分析流量沿网络传播过程中流量特征的变化和网络流量的统计行为,建立流量模型。

(王晓东)

wangluo chuliqu

网络处理器 (network processor) 面向网络和通信应用的嵌入式可编程微处理器。与通用嵌入式处理机不同,网络处理器专用于网络和通信协议的处理,因此它的体系结构相应地根据网络应用而变化。根据 ISO/OSI 模型的网络协议层次,网络处理器可细化为:链路层网络处理器、网络层(IP)及其上层处理器等。在速率方面,根据 SONET/SDH 层次大致可分为 OC-12/STM-4, OC-48/STM-16, OC-192/STM-64 等几个层次。相应的通信速率大致为 622 Mb/s, 2.5 Gb/s, 10 Gb/s 等。

发展简史

早期的高速通信和网络处理器大多由专用大规模集成电路(ASIC)来实现。专用集成电路有效率高、速度快、省电以及可靠等优点。这样的实现方式很适合成熟稳定的通信协议。其主要缺点有:①功能转移难,不易改动;②设计和投片成本高,不适用于小规模生产。

现场可编程器件(FPGA)是另一类常用的电路实现手段。当生产批量小时,现场可编程器件的综合成本比专用集成电路低。

随着因特网的飞速发展,网络处理器带来了通信设备设计方式的革新。第一代网络处理器产品在 1999 年—2000 年开始出现。最早的产品有 C-port C5, Intel IXP1200, Sitera, Agere 等。这些网络处理器大多采用片上系统(SOC)技术。它们大多集成了一个或多个精简指令集(RISC)的中央处理器(CPU)、高速片外存储器接口、用于表查找以及数据包分类和包队列缓存等专用硬件加速部件以及专用的通信接口的物理层或链路层(如以太网的 MAC)等。大多数产品设计时面对的应用为基本的 IP 协议交换功能,目标的速率多为 OC-48/STM-16 (2.4 Gb/s)。这些网络处理器的基本设计思想为利用运行在多个

CPU 上的软件来实现网络协议处理的基本控制流以及较为简单的数据处理功能,用集成片内的专门硬件单元来加速实现用软件所不易实现的实时处理功能(如 CRC 校验、高速查找 IP 路由表格、数据包分类、队列和缓存、包交换等功能)。此外,片上集成的专用链路层协议功能可方便地和系统中的相应物理层芯片连接,可减少外部逻辑电路或芯片的数量。由于第一代网络处理器的体系结构比较简单和理想化,在实际应用中用户所需的复杂实时处理功能往往难以达到,因此,在大多数通信系统中,成熟稳定的协议处理功能(如 IP 交换功能)常常还是采用专门的硬件集成电路来实现,网络处理器只提供一个软件可控制的部件以供实现额外的功能,起到补遗和提供可扩展性的目的。此外,网络处理器提供的链路层也大多局限在以太网接口,不便于处理更广泛应用于广域网中的 SONET/SDH、ATM、帧中继等协议。因此,网络处理器的片上系统集成的目标在很多应用中反而带来不便。

2001 年间出现了第二代网络处理器。其中的代表为 PMC-Sierra 和 Broadcom 等公司生产的基于 MIPS 指令集的精简指令集微处理器。这些网络处理器除了多个高速的 CPU 核以外还集成了千兆以太网接口以及其他标准的通信系统总线接口(如 SPI 3, SPI 4.2 等)。其目标是提供一个高性能的微处理器部件供网络系统中的控制层和数据层软件处理。这些处理器在处理网络路由协议和信令协议方面获得了广泛的应用。此外,由于它们采用了通信系统内部常用的总线接口,因而很容易和市场上已有的各种通信集成电路连接而构成复杂的系统。而且 MIPS 指令集保持了很好的软件兼容性,这类处理器因此易于编程。

2002 年出现了第三代网络处理器。主要代表为 Intel 公司的 Xscale 处理器。这些处理器集成了多个高速的 CPU 单元,其数量可以大于 64,其互联结构类似于传统的大规模并行处理(MPP)系统。在编程方面也与 MPP 系统类似,支持虚拟共享存储及消息传递编程模式。每个处理单元有私有的程序存储器和少量的局部数据存储。它们通过互联网络和外部高速存储总线以及通信外设和硬件加速单元交换数据和控制消息。这种高度并行的体系结构在单片上实现了单个处理器不能达到的极高处理速率(大于 20 000 MIPS)。这一代处理器还充分利用最新集成电路生产技术提供的大量的门资源来提高处理器的数量和片内高速存储量,而并非像第一

代网络处理器那样试图在一个芯片上实现系统集成,从而避免了把宝贵的硅片面积资源浪费在很少使用的外设单元上。此外,这些处理器也采用了通信系统中广泛使用的标准总线(SPI4),从而达到现有通信器件与不同厂家处理部件之间的方便互连。相应的处理速率可达到 10 Gb/s(OC-192/STM-64)以上。软件功能可覆盖第二、三层及其上层的协议处理。

硬件结构

网络处理器通常集成多个高速的精简指令集微处理器以实现一个高度并行的系统。CPU 之间由高速内部总线或互联交换网络相连。CPU 之间还实现了缓存的一致性协议,以便于编程。网络处理器中还通常集成了硬件加速单元以针对网络处理中常用的校验、查表、队列处理及流分类等功能。网络处理器还包括了标准通信总线,以便和外部通信器件互连。此外,网络处理器还实现了多个外部高速静态和动态存储器总线,从而消除网络处理中常常遇到的存储带宽的瓶颈。常见的总线有: Rambus, DDR-SDRAM, DDR-SRAM, QDR-SRAM, RDRAM 等。

发展趋势

网络处理器经过 3 代的发展,已经用在各种网络硬件设备中。其可编程性带来的多方面优点是其他技术所不能取代的。此外,网络处理器还把传统通信系统的面向硬件的设计方式逐步转向面向软件的设计方式,这带来了代码复用的优点,使得系统开发的智力投资可不断积累,从而减少系统开发的长期费用。未来网络处理器的发展趋势如下:

- (1) 单个 CPU 的性能不断提高。
- (2) 片内处理器数量和并行度提高。互连网络的速率也相应提高,以保证系统性能的可扩展性。
- (3) 开放式的指令集,合理方便的编程接口,并支持多机多线程。保证两代间的兼容性,以保护用户的软件开发投资。
- (4) 开放式标准总线的支持,保证网络处理器和其他通信器件的方便互联。
- (5) 高速高效的外部存储器总线,以保证内部算法不受存储器带宽瓶颈的影响。
- (6) 根据不同的应用场合,合理地集成专门的加速部件及通信单元。

(廖恒)

wangluo fuwu

网络服务(network services) 网络向它的高

层用户提供的数据传输方式和相关的处理方式。数据传输服务包括面向连接的服务和无连接的服务等。除了数据传输方式的服务之外,还有网络传输质量方面的服务,例如时延大小,抖动程度等。

服务也是分层的。第 n 层服务是第 n 层协议所提供的外部行为体现。服务规范被用来定义和描述第 n 层协议提供的服务,描述服务使用者和提供者之间交换作用的规则。

面向连接的服务在计算机开始通信之前,必须在通信网络之间建立一条端到端的通信链路,然后开始数据通信。待数据通信结束后,再终止这个连接。面向连接服务可分为三个部分:建立连接、数据传输和断开连接。

大多数网络系统为面向连接的服务提供数据流界面,例如 Internet 中的 TCP 协议提供调用接口套接字。使用这种界面,用户不用考虑如何划分数据分组或低层如何传输。用户只是把自己所要传输的数据看作一个流送给对方。

面向连接的服务又分为永久性连接服务和非永久性连接服务。非永久性连接服务在数据流传输之前,都要先进行连接,待连接成功后再进行通信。永久性连接则是在第一次数据流传输前进行连接。待该连接成功后,把相应的连接路径存入计算机,以后的通信中,不再进行连接。

无连接的服务的工作方式与邮电系统相似。无论何时,计算机都可以向网络发送数据包,而无需事先建立连接。与面向连接的服务不同,无连接的服务方式传输的每个数据包中必须包含目的地址,以便寻找合适的路由。由于无连接方式不需要接收方进行回答和确认,因此,无连接方式不能防止包的丢失、重复和失序。与面向连接的服务相比,无连接的服务占用网络资源少,处理开销小,发送消息快,但可靠性较低。Internet 中的 UDP 协议,以太网协议等都提供无连接的服务(参见用户数据报协议)。TCP 协议、ATM 协议等都提供面向连接的服务。

参考文献

张尧学等. 计算机网络与 Internet 教程. 北京:清华大学出版社,1999

(张尧学)

wangluo fuwu zhiliang

网络服务质量(quality of network services)

发送和接收信息的用户之间以及用户与传输信息的网络之间关于信息传输的质量约定。它包括用户要求和网络服务提供者的行为两个方面。用户要求指

用户在网络上进行多媒体通信时所要求的服务类型以及相应的传输性能和质量等。服务提供者的行为则指网络针对某一类服务所能提供和达到的性能与质量。

网络服务质量不是网络中某个个体或元素的行为描述,它涉及到用户与用户、用户与网络以及网络内部结点的整体行为。

服务质量控制涉及到用户与用户之间、用户与网络的各个元素之间的协调管理、资源分配等问题。

网络中引入服务质量控制的主要原因是为了传输实时的多媒体信息流和合理利用网络资源。目前 Internet 上传输的信息主要以数据流为主,而且主要是采用先来先服务的尽力而为的方式。这就阻碍了语音及图像等信息在 Internet 上的实时传输。

服务质量控制需要解决下述几类问题:

(1) 服务质量的分类与定义 异步传送模式(ATM)和 OSI 基准(参考)模型等都对服务质量的类型和参数进行了规范化定义。Internet 协会(IETF)也对服务质量的分类和参数给出了严格定义,制定了与服务质量相关的一系列提案和标准。其中,国际标准化组织(ISO)最早开始研究网络中的服务质量问题,并针对 OSI 基准(参考)模型,给出了每层协议所必须具有的服务质量。由于 OSI 基准(参考)模型未考虑到多媒体传输问题,其对服务质量的定义是不完整的。ATM 论坛把服务质量的定义引入了 ATM 交换机,并以此对各种用户应用和服务进行了分类。ATM 论坛把网络服务定义为 5 类,即恒定比特率服务(CBR)、实时可变比特率服务(rt-VBR)、非实时可变比特率服务(nrt-VBR)、自适应式比特率服务(ABR)和未指定比特率服务(UBR)。IETF 则从集成服务与区分服务的角度定义了服务质量的控制模型、实现框架和有关参数。另外,IETF 还定义了相关的资源预约协议。

(2) 准入控制 根据用户的服务质量要求和网络资源情况,控制使用网络资源的用户数量,从而保证正在使用网络资源的用户和享有高优先权的用户所需要的服务质量。

(3) 资源调度算法 网络是由各种软、硬件资源组成的,这些资源包括带宽、处理机、缓冲区、存储设备以及各种软件资源。如何根据用户的服务质量要求来调度和分配资源是服务质量控制所要研究解决的重要问题之一。

(4) 基于服务质量的路由 Internet 的路由算法是以最短路径法为基础的,即路由器对要在 Inter-

net 上互相通信的两台计算机计算它们之间的跳数,并以跳数最小的路径作为通信路径,但对于时延、跳变以及传输信息的种类等则未作考虑。服务质量控制机制要求路由器在进行路由计算时,按照用户的服务质量要求选择通信路径。

(5) 资源预约协议 为了保证用户获得所要求的服务质量,网络必须为用户预留出相应的资源,例如带宽和处理机处理时间等。资源预约协议被用来为用户预留资源。

(6) 服务质量协商与再协商 用户的服务质量要求有时很难与网络所能提供的服务质量一致。而且在大多数情况下,服务质量要求可以在一定范围内进行调整,例如每秒钟传输图形的帧数。服务质量控制机制提供用户和网络之间的协商机制,以及在信息传输过程中网络资源发生变化后的再协商机制。

服务质量是 Internet 进一步发展的关键问题和重大瓶颈之一。要在 Internet 上传输数据和音频、视频等信息,如果不对服务质量进行控制,仅靠增加带宽是不够的。端到端的服务质量控制机制正成为全世界网络界研究的热点问题。

参考文献

张尧学等. 计算机网络与 Internet 教程. 北京:清华大学出版社,1999 (张尧学)

wangluo gongcheng

网络工程 (network engineering) 计算机网络建设需要根据用户的需求及具体情况,结合现时网络技术的发展水平及产业化程度,经过充分的需求分析和市场调研,从而确定网络建设方案,依据方案有步骤、有计划实施的网络建设活动。网络工程建设是一项复杂的系统工程,一般可以分成以下几个阶段:

(1) 需求分析与网络规划 组网工程的第一步是进行需求分析,其任务是了解用户的具体要求,掌握用户目前的状况,提出工程的目标,制定系统概要设计书(参见网络规划)。

(2) 网络设计 网络设计是网络建设能否成功的关键环节。负责网络设计的工程人员既要了解系统集成的一般规律,又要理解计算机网络的体系结构、协议和标准,掌握计算机网络的技术、发展和趋势。这样才能根据用户的特点设计出符合建网目标的具体施工方案,指导网络工程的实施。

(3) 工程组织与实施 网络设计完成以后,就

要根据设计方案组织工程的实施。它可分成部件准备阶段、安装调试阶段、测试验收阶段及用户培训阶段。部件准备包括机房装修、设备订货、设备到货验收、电源准备和检查、网络布线和测试以及远程网络线路租借等;安装调试包括计算机安装和分调、网络设备安装和分调、网络系统调试、软件安装和分调、系统联调等;网络系统的测试和验收是保证工程质量的关键步骤,从内容上包括计算机及配套软件的测试、网络设备及配套软件的测试、布线系统的测试、应用软件和应用协议的测试以及计算机系统和网络系统的集成测试。通常在工程实施之前就要制定各项测试计划,测试完毕写出测试报告,以便作为验收和维护的依据。

(4) 网络运行与维护 网络系统的失效所造成的影响往往是惊人的,因此注意网络的日常维护,保证网络系统的稳定可靠,能够长时间高效率运行显得越来越重要。网络维护工作应该包括系统失效原因分析及后果分析、预防性维护、故障处理和系统的扩展几个方面。

参考文献

张卫,王能,俞梨阳等. 计算机网络工程. 北京:清华大学出版社,2004 (相士俊)

wangluo guanli

网络管理 (network management) 对计算机网络的配置、运行状态和计费等进行的管理。它提供了监控、协调和测试各种网络资源以及网络运行状况的手段,还可提供安全管理和计费等功能。随着计算机网络的规模越来越大、结构越来越复杂,不可能仅靠人工来进行维护和管理,网络管理软件已成为网络中一个必不可少的部分。

简单的局域网也需要管理。例如,Novell 公司的网络操作系统 NetWare 中就提供了用户管理、记账管理、工作站管理和服务器管理等功能。又如,许多用来组建局域网的智能集线器和交换机都提供对网络管理的支持。通过在网络中某个工作站上运行的网络管理软件,如 Netview,就可监视和控制网络配置的实时状态,记录和统计网络中各处的数据流量和差错情况。

在广域网中,网络管理更为必要,也更为复杂,通常建有专门的网络管理中心。有的网络管理中心还提供通信子网中路由控制的功能。某些大型网络允许设立多个管理分中心,由这些分中心各自实现对局部范围内有限功能的管理。例如,中国公用分

组交换数据网(CHINAPAC)就可以提供虚拟子网的业务功能,即允许用户利用该公用数据网的资源来组建自己的专用子网,并由用户自己的网络管理设备对专用虚拟子网内的用户自行管理。

由许多子网互连而构成的网际网络,更必须有网络管理机制才能有条不紊地运行。著名的 Internet(因特网)就是一个典型的例子。为了在因特网中交换网络管理信息,专门制定了一个简单网络管理协议(SNMP)。它在一个称为征求意见文献 RFC 1067 的文献中首先被定义,后来又经不断修改与完善,已得到了广泛的应用。

网络管理系统一般由管理信息库(MIB)、被管资源代理、网络管理软件和网管协议软件等部分构成。MIB 由一系列被管对象组组成。包括了被管对象信息的定义和信息本身。在 RFC 1066 中定义了基于 TCP/IP 的因特网管理所用的 MIB。被管资源代理一般运行在资源所在地,而网络管理软件则运行在称为管理工作站的计算机上,两者之间遵循专用的网络管理协议通过网管协议软件进行通信。被管资源代理从管理工作站接收管理协议操作,作用于 MIB 中的抽象的被管对象,并最终映射到实际的被管资源,如路由器的网络地址、交换机某个端口等。

按照网络管理软件和网络信息库是集中在网络内的一台计算机上,还是重复或分散在网络内不同的计算机上,可将网络管理分为集中式网络管理和分布式网络管理两大类。前者较简单且易于实现。一般小型的计算机网络,如局域网,大多采用集中式管理的方案。但是它也较为脆弱。一旦所在的计算机发生故障,就丧失了全部管理功能;分布式管理为了保持数据的一致性和动作的协调,实现起来较为复杂,但是可以提高可靠性。而且,若分布状况设计合理,则由于增加了并行度和分散了信道负荷,还可以提高管理的效率。

国际标准化组织在国际标准 ISO 7498-4 中为基于开放系统互连(OSI)的网络管理体系结构规定了框架。后来又陆续颁布了一系列有关 OSI 管理的其他国际标准。例如,

- ISO 9595: 公共管理信息服务定义;
- ISO 9596: 公共管理信息协议规范;
- ISO 10040: 系统管理综述;
- ISO 10165—1: 管理信息模型;
- ISO 10165—2: 管理信息定义;
- ISO 10165—3: 管理属性定义;

- ISO 10165—4: 定义被管对象的指南;
- ISO 10164—1: 对象管理功能;
- ISO 10164—2: 状态管理功能;
- ISO 10164—3: 关系管理功能;
- ISO 10164—4: 报警功能;
- ISO 10164—5: 事件报告管理功能;
- ISO 10164—6: 日志控制功能;
- ISO 10164—7: 安全报警功能;
- ISO 10164—8: 安全审计跟踪功能;
- ISO 10164—9: 对象和用于访问控制的属性;
- ISO 10164—10: 计费功能;
- ISO 10164—11: 工作负荷监视功能。

从以上 ISO 10164 标准的标题可以大致看出国际标准所规定的网络管理功能。国际电信联盟的电信标准部(ITU-T)也有和上述 ISO 标准等效的 X.700 系列建议。

概括地说,网络管理的功能主要有五个方面:

- (1) 故障管理 监视出错警报、发现故障、定位故障及从故障中恢复;
- (2) 记账管理 向合法用户提供网络资源,记录其使用情况并计费;
- (3) 配置管理 收集、显示和修改网络的配置参数,在网络结点或链路故障时选择替换路由;
- (4) 性能管理 监视、分析和控制网络交通及性能,保证服务质量;
- (5) 安全管理 维护网络中传输数据的完整性与可靠性,防止非法的入侵、窃听、篡改和冒用。

参考文献

1. 谢希仁. 计算机网络(第2版). 北京: 电子工业出版社, 1999
2. 高传善等. 数据通信与计算机网络. 北京: 高等教育出版社, 2000 (高传善)

wangluo guanli biao zhun

网络管理标准 (Network Management Standard) 用于对通信和计算机网络进行计算机辅助管理的标准。由于网络的应用越来越广泛,网络的规模越来越大,网络的结构越来越复杂,因此必然要依靠网络管理系统来辅助人们进行维护与管理。一个大型复杂的网络中具有不同类型的软、硬件,也可能出自于不同的供货厂商,要进行统一的管理,就要遵循某些统一的标准。网络管理标准涉及管理体系结构、网络管理协议和管理信息库等方面。**OSI 管理体系结构** (ISO 10164, ITU-T X.700 系

列)、电信管理网络(TMN, TTM - T M. 3000 系列)、公共管理信息协议(CMIP, ISO 9596, ITU - T X. 711)、简单网络管理协议(SNMP, RFC 1157)和管理信息库(MIB, RFC 1066 和 RFC 1213)等都是网络管理标准。

参考文献

Stallings W. SNMP, SNMPv2 and RMON. MA: Addison - Wesley, 1996 (高传善)

wangluo guihua

网络规划(network planning) 在用户需求分析和系统可行性论证基础上确定网络建设总体方案和**网络体系结构**的过程。网络规划直接影响到网络的性能和分布情况。一项**网络工程**能不能既经济实用,又兼顾长远发展,网络规划是重要的一环。

用户需求分析

需求分析可以采用自顶向下的分析方法,了解用户单位所从事的行业,该单位在行业内的地位及和其他单位的关系等。不同行业的用户,同行业内的不同单位,对信息网络的需求和它本身在信息网络中所承担的角色各不相同。不同角色的单位在进行网络规划时所采取的策略也不同。了解项目背景,有助于更好地了解用户单位建网的目的和目标。

对用户单位的建网目的和目标进行分析之后,应进行纵向的、更加细致的需求分析和调研,从而明确以下几个主要方面的情况:

(1) 地理布局 了解用户单位的建筑物布局,入网网站的分布情况。记录下述信息:

①网络中心(或计算中心)及各级设备间的位置;②用户数量及其位置;③任意两个用户之间的最大距离;④用户群组织,即在同一楼里或同一楼里的用户,尤其注意那些地理上分散,却属于同一部门的用户;⑤特殊的需求或限制,例如,网络覆盖的地理范围内是否有道路、山丘,建筑物之间是否有阻挡物,电缆等介质布线是否有禁区,是否已存在可利用的介质系统等。

(2) 用户设备类型 包括:

①终端——指没有本地处理能力的用户设备;②个人计算机——指具有本地处理能力的单用户或多任务个人计算机;③主机及服务器——具有本地处理能力的多用户设备;④模拟设备——电话、传感器、视频设备。

(3) 网络服务 包括:

①数据库和程序的共享;②文件的传送、存取;

③用户设备之间的逻辑连接;④电子邮件;⑤网络互连;⑥虚拟终端。

(4) 通信类型和通信量 通信类型有以下几种类型:

①数据;②视频信号;③声音信号。

不同类型的通信量用不同的度量。一般数据的通信量用平均的以及高峰时每秒传送的位数表示。视频信号的通信量用电视通道数表示,每个通道占6 MHz 带宽,声音信号则用欧拉数表示。

估计通信量比较好的做法是分析用户的网络应用,估计每个应用产生的通信量,再把各种通信量累计出结果。最后应把通信量都表示为每秒传输的位数。通信量的估计还可以通过对已有网络系统的调研得到。

(5) 容量和性能 网络容量是指在任何时间间隔内,网络能承担的通信量。网络性能一般用数据经过网络的响应时间或端一端延时表示。通常,当网络的通信量接近其最大容量时,响应时间就变长,网络性能就会恶化。网络规划者只有掌握了网络上将负担的通信量以及用户响应时间的要求后,才能选择网络的类型及其配置,以便更好地满足需求。

(6) 网络现状 如果要在已有的网络上规划建设新系统,那么了解用户单位现有网络的情况,尽可能在设计新系统时考虑旧系统的利用,这样既可保护用户投资,又能够使用户在系统的使用上有一个平滑过渡,节省培训时间和费用。

系统可行性分析

可行性分析是结合用户单位的具体情况,论证建网目标的科学性和正确性。通过可行性分析可以提出一个满足用户需求的网络体系结构,写出可行性报告。可行性报告大致包括以下几方面的内容:

(1) 可行性研究的前提 包括项目要求、项目目标、项目假定条件和限制条件、可行性研究方法以及对系统评价尺度等。

(2) 现状分析 分析目前用户的计算机使用状况,进一步说明组建新的计算机网络的必要性。

(3) 建议建立的网络系统方案 对网络体系结构的论述包括以下四个方面:

①传输 传输方式用基带还是宽带传输,通信类型及通道数,通信容量,数据传输速度。②用户接口 支持的协议,工作站类型,主机类型。③服务器类型,容量,协议。④网络管理能力 网络管理,网络控制,网络安全。

对于网络体系结构的描述,在可行性论证阶段应尽可能使用与厂家无关的功能术语。例如用单一基带通道来解决用户所需的多大的通信量,而不必决定选用标记传递还是争用访问方法。主要的是要说明所提出的网络结构是怎样满足用户需求的。网络结构中可包含多个网络或网络段。例如包含多个**局域网**,或者既有局域网,又有**广域网**。

(4) 可选择的其他方案。

(5) 投资与效益分析。

系统可行性的一个重要影响因素是造价,而这部分是要进行方案设计之后才能确定的。网络系统的方案往往不止一个,而且实施的效果和可靠性保证也不尽相同,用户的决策者可以从中选择出最佳方案。

(6) 社会因素 包括法律、制度和用户使用人员素质等。

(7) 结论 可行性报应给出研究结论,立即实施,不能实施或等条件成熟后实施等,并作出简要说明。

(王晓东)

wangluo hulian

网络互连 (internetworking) 将多个网络互连接以实现更大范围内的信息交换、资源共享和协同工作。称为网络互连的著名的 **Internet** 就是由成千上万个不同的网络互连而构成的国际网,或称为互连(联)网。**网络互连技术**包括**网络互连协议**和**网络互连设备**等方面。在网际网中,信息传输的源和宿可以在不同的网络中,在传输时还可以通过不同的中间网络从源出发到达目标,这就是网际的**路由选择**。路由选择时也要遵循相互认可的标准的**路由协议**。

(高传善)

wangluo hulian jishu

网络互连技术 (internetworking techniques)

将两个或多个**计算机网络**相互连接以实现信息交换和资源共享的技术。网络互连技术是计算机网络及其应用发展到一定阶段的必然产物。社会信息化对信息交换和资源共享的需要日益增长,范围也越来越广,这既包括应用领域的扩展,也包括地理区域的增大。形形色色的网络和产品不断涌现,如 **SNA 网**、**DECnet 网**、**分组交换网**、**同步光纤网**、**卫星网**、**ATM 网**和**无线网**等,而**局域网**种类就更多。不同的网络可适应不同的特定需要,而且各部门都希望对

自己拥有的网络具有控制和管理的能力,因此不可能构成一个类型单一的大型网络来满足社会各方面的需求。信息交换和资源共享问题只有通过网络互连技术将各种网络相互连接的方法来解决,著名的 **Internet** 就是网络互连技术成功应用的范例。

不同的计算机网络有不同的体系结构。它们所采用的网络协议、数据单元格式和长度、差错检验和恢复方法、流量控制措施、寻址方案、**路由选择**算法、管理和控制方式以及数据传输速率等都可能不同。在进行网络互连时,一般不应对互连在一起的任何网络的体系结构进行修改,而应通过网络互连技术来转换或适应这些差异。网络互连技术包括**网络互连设备**和**网络互连协议**等方面。

网络互连时,不是简单地直接相连,而是要通过各种网络互连设备来连接。利用网络互连设备可以进行被互连网络间差异的转换。按在**开放系统互连基准(参考)模型**(**OSI/RM**)中的哪一层上实现互连,可将网络互连设备分为**中继器**、**网桥**、**路由器**或**网关**。它们分别对应于物理层、数据链路层、网络层或更高的层次上实现互连。一般说来,互连的层次越高,要解决的差异转换问题也越多,网络互连设备也就越复杂。网络互连可分为**结节点级互连**和**主机级互连**两类。结节点级互连就相当于在网络层以下(包括网络层)的互连,而主机级互连则相当于在网络层以上更高层次的互连。此外,按被互连的网络类型还可将网络互连分为**局域网和局域网的互连**、**局域网和广域网的互连**或者**广域网和广域网的互连**三类。

类似于计算机之间的连网要遵循协议,计算机网络之间的互连也要遵循一些专门的网络互连协议。最著名的两个与网络互连有关的协议是 **Internet** 中采用的**网际协议**(**IP**)和原国际电报电话咨询委员会(**CCITT**),现国际电信联盟下属的电信标准部(**ITU-T**)的**X.75 建议**。

国际标准化组织(**ISO**)在制定**开放系统互连基准(参考)模型**初期未对网络互连的问题予以充分的考虑。后来,根据 **Internet** 的经验,在国际标准 **ISO 8648**中规定了网络层的内部结构,将网络层再进一步细分为若干个子层。其中,最上面的一个子层就是用于网络互连的子层。**IP**可看成是在该子层的协议。原 **CCITT**也专门考虑了各种**数据通信网络**之间的互连问题,并发布了与网络互连有关的**X.300 系列建议**。该系列建议包含有十几个建议。其中,还考虑了网络互连后的管理问题,即如何安排

网络间管理信息的传输问题。

参考文献

1. Perlman R. Interconnections. Second edition. MA: Addison-Wesley, 2000

2. 高传善等. 数据通信与计算机网络. 北京: 高等教育出版社, 2000 (高传善)

wangluo hulian shebei

网络互连设备 (internetworking equipment)

用来将两个或多个计算机网络或网络段连接起来的一种中间设备。若干个计算机网络需要互连时,一般都不能简单地直接相连,而必须通过中间的互连设备。有时,逻辑上虽是一个完整的网络,但由于受信号在媒体中传输特性的限制(例如,衰减只有在一定距离内才能不超过允许的限度等),物理上却必须划分为若干网络段,段与段之间也要通过网络互连设备来连接。

网络互连设备在开放系统互连基准(参考)模型(OSI/RM)中都看成是中继系统。按在 OSI/RM 的 7 层参考模型的哪一层上实现网间中继,可将网络互连设备分为中继路、网桥、路由器或网关四种类型。

中继器是在最低层,即物理层上实现互连的设备。通常用来连接同一网络中的不同网络段。例如,符合 ISO 8802-3 标准的 10 BASE 5 规范(参见以太网)规定每个网络段的长度不超过 500 m。超过 500 m 就要分段,而后通过中继器来连接。中继器能将从一个网络段中接收到的信号放大或再生后再向另一网络段发送,因此中继器有时也称为转发器。

网桥是在数据链路层实现中继的网络互连设备。它通常用于不同局域网间的互连。如符合 ISO 8802-3 标准的以太网与符合 ISO 8802-4 标准的令牌总线网之间的互连。

路由器是在网络层实现中继的网络互连设备。它不仅可用来互连局域网,还可用来互连局域网与广域网或实现广域网之间的互连。路由器与网桥的不同点是它具有路由选择和强得多的流量控制能力。

网关是在运输层以上(包括运输层)实现中继的网络互连设备。它的一个主要功能是实现被互连网络的不同协议之间的转换,即将从一个网络中接收的按该网络协议组织的报文转换成按另一网络协议组织的报文,再发送出去。因此,网关有时也称为

协议转换器。

参考文献

上海市科学技术委员会. 网络通信技术实用大全. 上海: 上海科学技术文献出版社, 1999

(高传善)

wangluo hulian xieyi

网络互连协议 (internetworking protocol)

为两个或多个计算机网络互相连接而设计的协议,即计算机网络间互相连接进行通信时有关数据格式及交互过程必须遵循的约定。计算机网络之间的互连必须遵循专门的网络互连协议,这和计算机之间连网时要遵循协议的道理是完全一样的。

早在 20 世纪 70 年代末,美国 Xerox 公司的 Palo Alto 研究中心(PARC)就研究了一种网络互连的结构,其核心思想是设计一种称为 PUP 的通用分组利用它在网络层对各个互连的子网进行中继。为满足 ARPA 网(参见 Internet)和各种计算机网络互连逐步发展成对 Internet 的需要,美国国防高级研究计划署(DARPA)于 1980 年公布了网际协议(IP)。它被收集进有广泛影响的 RFC 760 文档中。RFC 760 后来又两次修订为 RFC 777 和 RFC 791,同时也成为美国国防部的军用标准 MIL-STD-1777。

IP 是一个最著名的网络互连协议。Internet 中成千上万个计算机网络主要是约定遵循 IP 而互连的。IP 处于开放系统互连基准(参考)模型(OSI/RM)网络层的网络互连子层。IP 是按数据报方式工作的,也就是说在交换数据前不必预先建立连接,每个数据报在传输时都可独立地进行路由选择。它向高层提供的是无连接服务,没有差错控制、顺序控制和流量控制的功能。这些功能通常由更高层的协议,如运输层协议,来实现。在 IP 上面有一个面向连接的运输层协议,即传输控制协议(TCP)。以 IP 和 TCP 为核心,加上其他一些协议构成了 Internet 上现今广泛应用的一套协议,即 TCP/IP 的协议集。目前 Internet 上使用的是第 4 版网际协议 IPv4,并正在试验和逐步向下一个版本 IPv6 过渡。

另一个著名的用于网络互连的协议是 X.75 建议。X.75 是由原国际电报电话咨询委员会(CCITT)制定的用于连接两个 X.25 分组交换网的互连网协议。CCITT 原是国际电信联盟(ITU)的一个下属组织,现已更名为 ITU 的电信标准部(ITU-T),它制定的标准都称为建议。许多国家中都有作为公用数据网(PDN)的 X.25 分组交换网,例如我国的 CHINAPAC 就是中国的公用 X.25 分组交换数

据网。各个国家的公用 X.25 分组交换数据网之间都是遵循 X.75 建议互连的。

参考文献

1. Tanenbaum A. S. Computer Networks. Third edition. New Jersey: Prentice Hall, 1996

2. 上海市科学技术委员会. 网络通信技术实用大会. 上海: 上海科学技术文献出版社, 1999

(高传善)

wangluo jicheng fuwu

网络集成服务 (network integrated services)

在 Internet 上同时提供传统的传输服务以及实时传输的数据、音频、视频等具有不同性能要求的传输服务。它类似于电信领域的综合业务数字网 (ISDN), 是从 Internet 的角度对各种服务进行集成。它实质上是从端到端的行为开始, 到网络中各元素如何控制和实现这些行为, 提供用户满意的服务质量的总称。

在 Internet 中, Internet 工程任务部 (IETF) 定义专门的规范和数据单元描述不同类型的集成服务。例如, 集成服务类型的取值范围在 RFC 2210 中被定义为 [1, 254]。IETF 定义了尽力而为服务、保证型服务和控制负载型服务等三种类型。尽力而为服务指传统 Internet 使用的先来先服务模式。保证型服务要求用户清楚地描述应用需求, 并指明实现机制。另外, 支持保证型服务的网络路径中的各个元素, 包括路由器和交换机等都必须支持保证型服务。否则, 该服务将会因为其中某个元素带来的延迟而失败。控制负载型服务是一种端到端的控制行为, 提供这种服务的网络使用户感到网络是在很轻负载或很大容量的条件下运行。用户的要求可能被延迟或得不到满足, 但这种延迟和得不到满足都是用户可以忍耐的。在具体实现上, 为保证控制负载型服务的条件得到满足, 用户将首先给提供控制负载型服务的网络一个所需流量的估计值, 然后, 网络中各元素将验证自己是否具有足够的资源为用户提供服务。如果有的网络元素不具备相应的服务能力, 则系统将其反馈给用户, 并进行协商调整。

与集成服务相对应的服务质量控制机制是准入控制机构、资源预约协议以及相应的调度算法等。实现集成服务要求网络中各元素具有按照集成服务要求的资源控制能力和管理能力, 这个要求较高。目前 Internet 上的集成服务仍处于研究过程中。

参考文献

张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999 (张尧学)

wangluo jisuan moshi

网络计算模式 (network computing mode)

把地理位置上分布的多个计算资源通过计算机网络在逻辑上组织成一个集中的计算资源的方式。

网络计算模式从 20 世纪 70 年代开始, 经历了从分时共享模式 (主机体系结构) 到资源共享模式 (文件共享体系结构), 再到客户-服务器模式的演变 (参见客户-服务器计算)。随之, 网络计算的可用性、可伸缩性、互操作性及可扩展性也在逐渐增强。而网络计算模式的最重要的特点是它的可扩展性, 它能够在不中断系统的情况下很快地适应系统的变化, 例如系统负载的波动和某些结点的故障等。

网络计算模式的发展也促进了网络运行环境的发展, 例如支持客户-服务器模式的一致通信环境和支持分布计算的分布式计算环境等。反之, 网络运行环境的发展也产生了新的计算模式, 如在万维网基础上发展起来的浏览器-万维网-数据库模式。

参考文献

胡道元. 计算机局域网 (第三版). 北京: 清华大学出版社, 2002 (王勇 相士俊)

wangluo qufen fuwu

网络区分服务 (network differential services, diffserv, DS)

除了提供传统的传输服务外, 通过对数据分类以及利用应用本身的可适应性等对实时传输的数据、音频和视频等不同性能要求的应用提供可扩展的定性传输服务。

为了解决网络传输的服务质量控制问题 (参见网络服务质量), Internet 工程任务部首先提出了集成服务模型。但要从现行的尽力而为传输模式升级到集成服务体系结构 (参见网络集成服务) 仍然存在许多困难。其中最主要的原因之一在于许多网络结点所采用的技术并不支持服务质量的实现, 或缺乏合适的信令支持, 从而削弱了网络提供端到端保证的能力。另一方面, 由于集成服务体系结构的实施粒度比较细, 所有的服务与控制都是针对数据流进行的, 因此提供强服务保证的同时也带来了实现复杂、开销大和可伸缩性差等问题。

基于以上原因, 研究人员尝试从另一个角度来考虑服务质量控制问题。例如, 通过对数据分组进

行分类以及利用应用本身的可适应性来满足各类服务的质量要求。由于只有小部分应用需要很强的服务质量保证,而这种保证可以通过提供足够高的带宽来满足峰值流量来达到,因此仅做粗略的优先级分类就足以保护它们。这些想法构成了区分服务的基本思想。

区分服务体系结构基于比较简单的模型:当数据流进一个网络时,它的分组在边缘设备中被分类标记,从而可以将各个分组归入各种具有相同码值的分组集合。每种分组集合通过一个单一的 DS 码值来标识。在网络核心中,路由器根据分组的 DS 码值采用相应的“每跳行为(PHB)”来转发分组,不同的 PHB 使得分组在通过路由器时具有不同的处理优先级。PHB 被定义为“在某一个 DS 兼容节点上对通过一条链路的具有相同 DS 码值的分组集合所施加的外部可见的转发行为”,它是区分服务的核心功能。PHB 是网络结点在不同汇聚流之间分配缓存和带宽资源的具体手段,它主要通过缓存管理和调度机制来实现。一组相关的 PHB 代表了区分网络的一类服务水平,通过为业务流标记 PHB 码值,就可以使这个业务流得到相应的服务。

区分服务功能由网络结点中许多功能元素实现,包括 PHB 集、分组分类功能、流量调节功能(测试、标记、整形和监控)等。

区分服务的显著特点是可伸缩性强,因而可以在大型网络中使用。可伸缩性强是通过将大量的复杂性工作放在在边界设备中完成来达到的。边界设备降低业务流的速率并减少业务流的数目,同时使服务基于汇聚流而非基于每个微流的粒度,从而使核心路由器的工作仅限于转发汇聚业务流,这样做有利于以后的扩展。

参考文献

1. Blake S, Black D, Carlson M, Davies E, Wang Z, and Weiss W. An Architecture For Differentiated Services. Internet RFC 2475, Dec. 1998
2. Nichols K, Blake S, Baker F and Black D. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. Internet RFC 2474. Dec. 1998

(王晓春)

wangluo ruqin jiance

网络入侵检测(network intrusion detection)

基于入侵者的行为不同于合法用户的行为,通过可以量化的方式对入侵行为进行检测和反应的网络

安全工具。

网络入侵检测的作用包括:

(1) 如果一个入侵行为被迅速地检测出来,就可以在任何破坏或任何数据泄密发生之前将入侵者识别出来并驱逐出去。虽然检测不能足够快地对入侵者采取先发制人的措施,但入侵行为越早被检测出来,破坏的程度就会越小。

(2) 一个有效的入侵检测系统可以作为一个阻碍物,用来防止入侵。

(3) 可收集有关入侵技术的信息,用于加强入侵防止机制。

用于入侵检测的技术有:

(1) 检测统计上的反常 在一段时间内对与合法用户行为有关数据进行收集,然后对所观察的行为用统计学的方法进行测试,以确定该行为是否是合法用户的行为。包括:①阈值检测:这种方法独立于用户,定义不同事件发生频率的阈值。②基于轮廓的检测:找出每个用户的行为轮廓,用来检测该用户单独账号行为的改变。

(2) 基于规则的检测 基于规则的技术通过观察系统中的事件,并且应用一个决定给定活动模式是否可疑的规则集来检测入侵行为。可以把所有方法分成集中于异常检测或渗透标识两个方面。①异常检测:制定用于检测给定行为与以前所使用的模式之间的偏差的规则。②渗透识别:用来搜索可疑行为的专家系统方法。

统计方法试图定义通常的或期望的行为,而基于规则的方法试图定义合适的行为。两种方法适用于不同的攻击类型,在实际操作中,有时需将两种方法组合起来应用。

(胡道元)

wangluo ruanjian

网络软件(network software) 计算机网络环境中,用于支持数据通信和各种网络活动的软件。目的在于实现分散系统之间的互连、互通和互操作功能;提供信息传输、资源共享、电子信函、远程操作、跨网协同工作等服务;支持网络系统和网络应用系统的运行、开发、检测、维护、管理等工作。网络软件主要包括通信软件、网络协议软件、网络应用系统、网络互连软件、网络支持软件等。

通信软件是支持用户程序与远程终端或计算机进行有效、可靠通信的软件。通常由线路缓冲区管理程序、线路控制程序和报文管理程序组成。线路缓冲区管理程序承担线路缓冲区的分配、使用和回收等管理工作。线路控制程序按照给定的传输控制

规程,控制线路的接续与断开,实现信息的发送与接收等工作。报文管理程序负责报文的接收、发送以及系统启停、收发日志、差错控制等日常维护管理工作等。

网络协议软件是网络软件的重要组成部分,通常按照选定的协议层次模型采用分层方式进行组织。例如,国际标准化组织(ISO)所建议的**开放系统互连**(OSI)基本模型,由物理层、链路层、网络层、传送层、会话层、表示层和应用层7层协议组成。又如,国际上最大的互联网因特网采用传输控制协议/网际协议(TCP/IP)模型,它由物理层、网络接口层、网际层、传送层和应用层组成。每层协议软件通常由一个或多个进程组成,其任务是完成相应层所规定的协议功能和上下层接口的服务功能。协议进程的实现梗概如图1所示。当请求某协议进程工作时,监督程序将相应协议进程激活。于是,协议进程就从相应队列中取一协议元素,然后检查该元素是否是合法元素。如果不合法,则转去进行错误处理。如果是合法元素,则进一步分析协议元素的类型,并完成相应的处理工作。重复上述过程,直至相应队列空时转去等待新的工作请求。

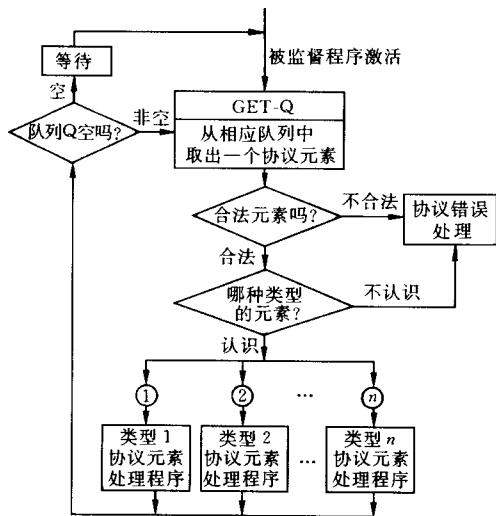


图1 协议实现示意图

网络应用系统有通用和专用之分。通用系统适用于比较广泛的行业或领域,如数据收集系统、数据转发系统、数据库查询系统等。专用系统一般只适用于特定的行业或领域,如银行核算、铁路控制、军事指挥等。

网络互连软件是用于实现网络之间相互连接的软件,通常安置在网络互连设备之中,按照支持

的互连协议层次来分,有中继器(物理层)、网桥(链路层)、路由器(网络层)、网关(传送层以上)等。其主要任务是在互连网络之间实现信息帧传输、数据格式转换、网际路径选择、相关协议转换等工作。

网络支持软件是用于支持网络软件的开发、检测、维护和管理工作的软件,是协议工程的重要研究内容之一。协议工程的主要任务是采用软件工程的方法,结合网络领域的特点,研究适用于网络协议软件的开发、维护过程及其相关技术。其研究内容包括:协议形式化描述技术及其描述语言,协议软件自动生成技术及其开发、维护工具,协议一致性测试技术及其测试工具等。

参考文献

1. 曹东启,刘福滋. 计算机网络软件基础. 北京:人民邮电出版社,1984
2. Tanenbaum A S. Computer Networks. New Jersey: Prentice Hall, 1988
3. 胡道元. 计算机网络工程指南. 北京:电子工业出版社,1993

(曹东启)

wangluo sheji

网络设计(network design) 根据网络规划及总体方案,对网络体系结构、子网划分、逻辑网络组成及网络技术和设备选型进行工程化设计的过程。

网络设计原则

网络设计过程中为使方案可行且能保护用户的投资,要注意遵从以下设计原则:

- (1) 成熟性原则 要采用成熟的技术,选用成熟的产品;
- (2) 开放性原则 要遵循国际、国内及相关行业的标准,采用开放技术、开放的体系结构、开放的系统组件及开放的用户接口,保证与其他系统良好的互操作性;
- (3) 安全可靠原则 稳定可靠,具有高的平均无故障时间和平均无故障率,提供容错设计,支持故障检测和恢复,可管理性强;
- (4) 先进性原则 在成熟性的基础上,尽量采用先进技术、先进的设备、先进的开发工具及先进的设计思想。但要注意实用性,以获得较高的性能价格比;
- (5) 完整性原则 实现优化的网络设计,既要求安全的数据管理、高效的数据处理,还要求友好的用户界面;
- (6) 可扩展原则 既能满足用户单位在入网机

器数量上的增长需求,又能满足用户因增加新的业务、新的应用而引起的对带宽增加的需求,能够在规模 and 性能两个方向上进行扩展。

网络设计的内容

网络设计的内容大致包括网络体系结构的确定、逻辑网络设计和网络技术与设备选型等。

(1) 确定网络体系结构

网络体系结构需根据用户的计算机及网络的应用水平、业务需求、技术条件及费用预算等进行恰当和合理的选择。

网络体系结构包括功能分层和各层通信所遵守的协议。现实可供选择的协议很多,TCP/IP 已成为网络通信协议事实上的国际工业标准(参见 TCP/IP 协议集),而 ISO/OSI 是公认的网络通信体系结构的国际标准(参见开放系统互连基准(参考)模型)。一个现代化的网络要适应多厂商、多协议和操作性的要求。

子网划分在很多情况下是必要的,例如,同一网段上的网络结点不能太多,否则会引起效率下降;不同部门间的信息需要隔离;某些特殊用户需要不同的网络带宽;需要多种网络操作系统等。可以根据具体情况按部门或按任务等决定子网划分的方式和方法,以求获得最佳的网络服务。

(2) 逻辑网络设计

逻辑网络设计包括逻辑网络拓扑结构设计、网络性能设计、网络地址分配与命名、选择桥接、交换和路由选择协议以及网络安全和管理策略等。

网络拓扑结构设计 网络拓扑结构说明的是网络的几何形状。在设计中,要确定网段和互连结点,明确网络大小和范围,以及所需要的网络互连设备类型。

从逻辑上讲,设计大型网络时可以从中心开始把通信网络划分为核心层、分布层和接入层。核心层的任务是为其他两层提供优化的数据传输功能,它由一个主干网组成;分布层提供基于统一策略的互连操作,定义了网络边界,可以对数据包进行复杂的运算,由路由器和交换机组成;接入层的主要目标是为最终用户提供对网络访问的途径,接入层网络设计也应考虑 Internet 的接入,由集线器、交换机和其他设备组成。

网络性能设计 网络性能主要指分组转发速率、吞吐量、分组丢失率、出错率、事务处理速率、响应时间、延迟时间、数据传输速率以及信道利用率等。网络性能设计的目标是使网络系统能满足用户各方面的要求。在设计阶段需要尽可能避免出现性能瓶颈,根据应用数据流的特点,设计性能监控和优化机制。

网络的可靠性和容错性能在一些重要场合的网络中是最为重要的性能指标。如国防、交通和金融证券部门要求网络能够适应复杂环境,长时间不间断地运行,系统即使发生故障也能继续运行下去等。在一般具有一定规模的网络中,也要求网络的核心层和关键设备具有一定的可靠性,使得整个网络系统能够平稳地运行。提高网络系统可靠性的方法除了保证系统本身的质量和规范的管理外,网络设计中主要是设计冗余部件,即构建网络系统的备份体系,增强系统的容错能力。具体的容错能力设计又可分为硬件容错、软件容错和线路容错。在信息系统中,完整的备份体系还应该包括运行环境备份、业务数据备份、备份策略和恢复方案。

网络地址分配与命名 网络拓扑结构确定后,就应进行网络站点设计,进行网络地址分配和命名。
①网络站点设计 网络站点是网络的基本元素。网络站点设计要从其功能、性能和标准几方面去考虑。网络站点可分为端站点和中继站点。网络的端站点构成了网络的资源子网,提供用户可以共享的应用资源,如各种类型的服务器、微机、外部设备、系统软件及应用软件等;网络的中继站点和通信线路一起构成网络的通信子网,为端站点提供通信服务。
②网络地址分配与命名 大量的网络应用都建立在 Internet 上,因此为 Internet 上互连的站点、服务器分配地址也是网络规划和设计的一个必要步骤。在设计过程中可以对多种资源命名,包括路由器、服务器、主机及打印机等。

网络安全与管理策略设计 通过对网络的全面了解,按照安全策略的要求及风险分析的结果,整个网络系统的安全设计可分为物理安全、网络安全和信息安全。物理安全是整个网络系统安全的前提,保证计算机网络设备、设施以及其他媒体免遭环境事故、人为操作失误等灾害,以及各种计算机犯罪行为的破坏。网络安全设计应该从系统安全、系统与网络的安全检测、访问控制、审计分析、反病毒、网络运行备份与恢复应急措施等方面去考虑。信息安全则从用户信息的角度提高网络的安全性。

网络管理设计 首先要确定网络管理目标,即用户对性能管理、故障管理、配置管理、安全管理及记账管理等方面的需求及实现的可能性。其次,确定网络管理机构,包括:网络管理设备,即收集和储存管理信息的网络结点,可以是路由器、服务器、交换机等;网管代理,即驻留在管理设备上的网络管理软件。再者,网络管理设计要确定网络管理的工具和协议等。

(3) 网络技术与网络设备的选型

无论是**局域网**还是**广域网**都有很多种类型,技术上也有很大不同;组成一个网络系统需要各类设备,有电缆连接器件,也有网络连接设备,如**集线器**、交换机、服务器和电源等。但其选择原则都是相同的,一要符合用户应用的需要,二要选择性能价格比好的产品。

到目前为止,大部分计算机网络需要使用广域网接入 Internet,而**宽带接入技术**已有了可喜的发展,网络设计阶段需要从整体的目标出发对接入技术进行选择,规划单位内部网络和服务商的广域网的连接方式。最重要的是网络带宽、可连接性、地址的识别和转换、互操作性以及安全性。

总之,选型的主要任务是选择网络布线系统(参见**结构化布线系统**)、物理层和数据链路层协议以及网络互连设备等。

参考文献

1. 胡道元主编. 网络设计师教程. 北京:清华

大学出版社,2001

2. 张卫,王能,俞梨阳等. 计算机网络工程. 北京:清华大学出版社,2004 (王晓东 相士俊)

wangluo shipeiqi

网络适配器 (network adapter) 把网络结点连接到通信媒体上使之与网中其他结点进行通信的一种接口部件。它往往以插卡的形式配置在结点机上,故又称**网卡**。网卡是一种选件。

网络适配器分为**局域网**中使用的和**广域网**中使用的两大类。在局域网中,常用的有**以太网**、**令牌环网**和**光纤分布式数据接口**三种网络适配器。广域网中则以 X.25 公用数据网的网络适配器较为常见。

图 1 为 10 BASE 2 的网络适配器连接图。在图中,若干台 PC 上安装了以太网适配器,由同轴电缆连接,形成 10 BASE 2 型以太网。

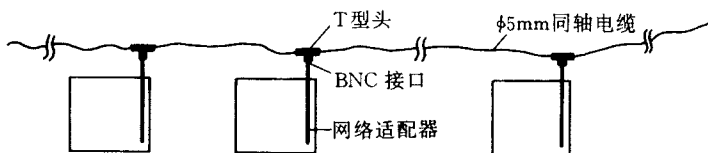


图 1 10 BASE 2 的网络适配器连接图

网络适配器通常实现**开放系统互连基准(参考)模型(OSI/RM)**的最低两层,即数据链路层和物理层。下面以以太网为例说明网络适配器实现的功能。

(1) 发送和接收 发送时,把结点要发送的数据在网络适配器中形成帧;接收时,分解出从媒体上接收到的帧中的数据,然后传送给结点。

(2) 媒体访问控制 此功能可解决各结点往媒体上发送信息时发生碰撞而引起的问题。解决的办法是:或者发送时避免碰撞;或者发生碰撞后再重新发送,直到发送成功。

(3) 数字信号的编码及译码 发送时把一般的二进制数字信号转换成媒体上传送的具有特殊编码方式的数字信号,以达到接收方能区分 0 和 1 代码以及实现收发双方时钟同步;接收时把从媒体上接收的特殊编码信号转换成一般二进制数字信号,并分离出时钟信号。

(4) 与媒体的物理连接 发送信号时,网络适

配器驱动媒体以使信号在媒体上传输,达到规定的有效距离;接收信号时,把媒体上衰减与畸变的信号进行放大和整形,然后进入网络适配器内部。

(张公忠)

wangluo shuju danyuan

网络数据单元 (network data unit) 网络中一次传输的数据单位。网络数据单元包括协议数据单元(PDU)和服务数据单元(SDU)。SDU 是相邻层实体间传送信息的数据单元,PDU 是对等实体间传送信息的数据单元,它是 SDU 加上同层协议控制信息(PCI)所形成的数据单元。SDU 即可以作为一个 PDU 传输,也可以在同一层次中被分成几段后,每段加上一个 PCI 组成新的 PDU 传输。

PDU 是通信协议处理的数据单元。不同层的协议具有不同的 PDU,例如 IP 报文是网络层的 PDU,而以太网帧是网络层的 PDU。同样,不同层的

SDU 也是不同的。

$n-1, n, n+1$ 层数据单元 SDU 与 PDU 的关系如图 1 所示。 $(n+1)$ PDU 是借助 (n) SDU 传到 n 层的, 此时 (n) SDU 就相当于 n 层的用户数据, 对它加上 (n) PCI 后便构成了 (n) PDU。这里, $(n+1)$ PDU

似乎等同于 (n) SDU, 实际上, 两者往往是不同的。有时, 发送方实体需要将数个 $(n+1)$ PDU 拼接成一个 (n) SDU, 而在接收方对等实体需进行一个 (n) SDU 分割成数个 $(n+1)$ PDU 的操作。

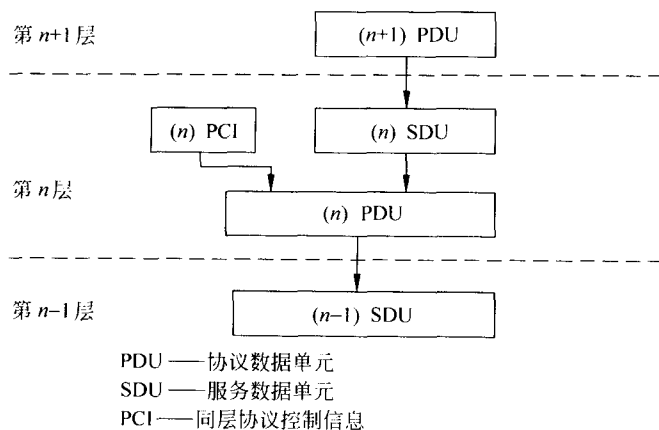


图 1 SDU 与 PDU 之间的关系

在开放系统互连基准(参考)模型的对等实体间的通信过程中, 两个 $(n+1)$ 层实体的通信要通过它们的低层(即 n 层)利用 n 层协议建立一种逻辑联系, 成为 (n) 连接。 n 层的对等实体的通信又借助于 $(n-1)$ 连接, 这样的过程一直递归到互连的传输媒体为止。在释放连接阶段, (n) PDU 释放 (n) PCI 后便构成了 (n) SDU, 然后传到 $n+1$ 层, 也就是 $(n+1)$ PDU。

报文、分组和帧均可视为网络数据单元。

参考文献

1. 张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999

2. 杨心强, 邵军力. 数据通信与计算机网络. 北京: 电子工业出版社, 1998 (张尧学 相士俊)

wangluo shujuku

网络数据库 (network database) 适用于计算机网络环境的数据库。它运行在计算机网络环境中, 是可供多个事务同时共享数据的数据库系统。

网络数据库系统是从单机、单用户数据库系统的基础上发展起来的, 与单机、单用户系统比较, 网络数据库系统具有如下特点:

(1) 具有并发控制功能, 能保证数据一致性。

在网络数据库中, 可能出现多个事务并发操作

某个数据的请求。必须通过对并发请求的控制来保证运行的事务能正确地执行和获得正确结果, 并确保数据完整性。在数据库管理中, 实现并发控制常用的方法是加锁。

(2) 具备数据安全性处理。

网络数据库中的资源可供多人在不同地点操作, 因此在提供方便的同时, 有可能引发数据失密、丢失、出错等情况。为此, 现有数据库系统都提供了一定的安全保密措施, 最常用的是口令验证。

(3) 数据处理能力增强。

在网络数据库中, 由于有众多的用户对资源共享, 使需要处理的数据量大大增加, 这就要求它比单机系统具有更好的数据处理能力, 以保证对各事务的及时响应。

比较流行的网络数据库系统有 Oracle、Sybase 和 Novell Btrieve 等。随着网络技术的发展, 网络数据库技术必将得到极大的发展。

参考文献

1. Date C J. An Introduction to Database System. 6th edition. Reading Addison Wesley Publishing Company, 1995

2. 萨师煊, 王珊. 数据库系统概论(第二版). 北京: 高等教育出版社, 1991 (王晓春)

wangluo tixi jiegou

网络体系结构(network architecture) 计算机之间相互通信的层次、各层次中的协议和层次之间接口的集合。**计算机网络**的设计是按高度结构化方式进行的。为了减少网络设计的复杂性和提高网络的可靠性,以及为了增加网络系统的开放性和互操作性,几乎所有的计算机网络都按层的方式组织和设计协议。层是指网络体系结构中功能划分明确的一个子部分。

不同的计算机网络具有不同的体系结构,其层的数量、各层的定义、内容和功能以及各层之间的接口都不一样。在任何网络中,层都是为了向它相邻的上层提供特定的服务而设置的,每一层如何实现协议和服务的具体细节都对上层屏蔽。这样,网络的体系结构就能做到与具体的物理实现无关,哪怕连接到网络中的主机和终端的型号和性能各不相同,只要它们共同遵守相同的协议就可以实现互相通信和互相操作,从而构成开放的网络系统。

网络中一台计算机和另一台计算机的通信必须在同一层次上进行。通信所用的规则和顺序以及所传递消息的格式构成该层的协议(参见**网络协议**)。计算机网络中的数据传送方式不是从发送方的第 n 层直接传送到接收方的第 n 层,而是每一层都把数据和控制信息传递给它的下一层,直到物理传输介质。接收时,则是每层从它的下一层接收相应的数据单元,并去掉与本层有关的控制信息之后把剩下的数据交给它的相邻高层。

各个相邻层之间都有相应的界面,该界面定义下层向上层提供的各种服务和有关使用这些服务的操作和响应。为了保护这些操作和响应的功能完整性,人们把它们设计成在执行过程中不允许中断或不允许并发执行的原语。网络设计者在设计网络体系结构时,首先必须要定义每一层所要完成的功能集合,然后定义上下层之间的接口界面,最后才设计为完成所需要功能与服务的协议。清晰的界面和具有明确含义的功能集合不仅使协议的设计和实现变得容易,而且使得在相同层中用一种不同的协议实现现代代码代替另一种协议实现现代代码成为可能,因为只要这两种实现现代代码能向上层提供相同的服务即可。

层的划分必须适当。层次太多会造成系统处理时间增加和包头长度增加,从而系统开销增加。这在那些要求高速传输的网络中是不允许的。但是,层次太少又会造成每层的功能不明确,相邻层间界面不易确定,从而使得协议的可靠性降低。大部分

网络体系结构的层次为 4~7 层。

首先提出计算机网络体系结构概念的是美国国际商用机器(IBM)公司。IBM 公司于 1974 年提出了系统网络体系结构 SNA。1978 年国际标准化组织 ISO 提出了**开放系统互连(OSI)基准(参考)模型**,并陆续推出了有关协议的国际标准,从而确立了 OSI 网络体系结构。这为不同制造商的计算机和不同计算机网络的互连提供了依据,因为只要这些计算机和网络遵守 OSI 体系结构,它们就能够互相连接和互相通信,并能够互相操作。

OSI 网络体系结构在很大程度上模仿了 SNA 网络体系结构。SNA 网络体系结构被分为七层。这七层是:①物理链路控制层:负责在计算机之间传递实际的原始比特流。②数据链路控制层:它以对上层透明的方式,把原始比特流组成帧,检测传输错误并进行恢复。SNA 体系结构的数据链路控制层中使用最多的是同步数据链路控制协议(SDLC)。SNA 体系结构在该层也支持高级数据链路控制协议(HDLC)和权标环局域网协议等。③路由控制层:它在通信源端和目的端之间建立一条逻辑通路。④传输控制层:它的任务是创建、管理和取消传输连接。SNA 体系结构不支持无连接通信。⑤数据流控制层:承担出错恢复等工作。⑥网络寻址单元服务层:该层向相邻高层的用户进程提供表示服务、会话服务与网络总体管理有关的网络服务。⑦用户进程:它是最高层。物理链路层为最低层。低层为相邻高层提供服务。数据包从上而下(除了物理链路层之外)进行发送传输,从下而上进行接收处理。每层协议都把相邻高层送来的数据包作为数据信息,并加上该层自己的控制报头之后转交给相邻低层。

ARPANET 是美国国防部支持开发的网络,现已演变为连接世界大部分地区的几千万台主机和 2 亿以上用户的 **Internet**。该网络使用 TCP/IP 协议,其体系结构只有①物理层、②链路层、③网际层、④传输层 4 层(参见**计算机网络**)。

计算机网络体系结构是一个不断变化发展的概念。当前,以 TCP/IP 协议(参见**TCP/IP 协议集**)与层次划分为主的网络体系结构是计算机网络体系结构的主流。但是,异步传送模式(ATM)等为主的 B-ISDN 体系结构、主动式网络体系结构等都在发展和研究中。

参考文献

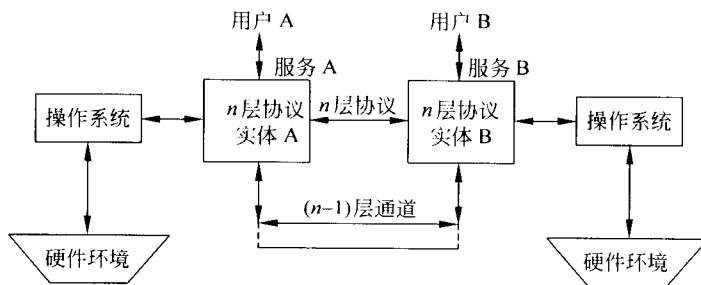
张尧学等. 计算机网络与 Internet 教程. 北京:

清华大学出版社, 1999

(张尧学)

wangluo xieyi

网络协议 (network protocol) 计算机网络和分布式系统中互相通信的对等实体间交换信息时必须遵守的规则集合。这里, 对等实体是指在计算机网络体系结构中处于相同层次的通信进程。协议具有和计算机语言几乎完全相同的定义, 即协议为传输的报文定义严格的格式 (语法) 和传输顺序 (文法), 而且, 协议还定义所传输报文的词汇和这些词汇所表示的意义 (语义)。

图1 n 层协议的通信环境

用户要求有: ①连接管理 包括是否需要建立和指定对等实体的通道连接以及连接的拆除等。②通信方式 指两通信的对等实体之间采用同步模式还是异步模式交换信息, 信息是采用单工还是半双工或全双工方式传送 (参见串行传输) 以及是采用点对点通信方式还是广播方式交换信息等。③通信对象 即通信地址。

通道性质指对等通信实体的连接方式和传输可靠性, 包括两通信实体独占一条连接通道, 几个实体共享一条连接通道以及任何通信实体都占有固定连接通道的无连接方式等几种。传输可靠性指通道是否为 n 层协议提供无差错传输。

协议运行时的操作系统及硬件环境包括操作系统的实时性, 缓冲区管理方式以及 CPU 速度, 总线方式等。与这些部分有关的协议往往与实现密切相关。

(2) 协议提供的服务 n 层协议所提供的服务是 n 层协议的外部行为体现。用户一般用服务规范来定义和描述 n 层协议所提供的服务。另外, 服务规范定义服务使用者和服务提供者之间的相互作用的规则, 也就是服务提供者向服务使用者提供服务的顺序。 n 层协议所提供服务的种类很多。例如,

(1) 协议的通信环境 网络协议可被抽象成一个层次模型。在层次模型中, 低层协议为相邻的高层协议提供服务, 高层协议调用相邻的低层协议提供的服务完成对等实体之间的信息交换功能。两个协议对等实体通过其低层协议构成一个通道。从而, 对于一个 n ($n \geq 2$) 层协议来说, 其低层协议构成 $(n-1)$ 层通道。另外, n 层协议还为本层协议的用户 (简称 n 层用户) 提供服务。用户要求、通道性质以及 n 层协议运行时的操作系统和硬件条件等构成了 n 层协议的通信环境, 如图 1 所示。

连接管理服务, 发送和接收数据包服务等。服务使用者根据服务规范, 按照不同的认可方式认可这些服务。

(3) 词汇表 词汇表定义 n 层协议中所使用的消息以及它们的意义。例如, “ack” 可被定义成接收方正正确接收到数据单元后的应答, “nak” 则可被定义成未正确接收到数据包或传输出错 (未到达接收方) 时的应答。

(4) 消息的编码格式 消息的编码格式是协议的语法定义, 它包括数据长度、控制信息长度以及每个域的定义等。

(5) 时序、规则和过程 时序、规则和过程是网络协议中最复杂、最关键的部分。它们规定用什么样的方法和算法去完成服务规范所定义的协议功能。

协议的功能除了包括连接管理、通信方式管理、协议数据单元的发送和接收以及装配和拆卸等之外, 还包括数据单元的编码和解码、分解和组合, 以及流量控制、拥塞控制、发送顺序控制、发送速度控制和出错处理等。协议的规则和过程被用来完成这些工作。

协议必须从语法和语义上严格定义协议中消息

交换的时序、规则以及有关过程等,否则将造成协议功能和用户要求服务规范不一致的局面。

由于协议的复杂性,在开发协议软件之前必须用自然语言(英语或中文)首先描述出协议的上述各项内容,以便使用该协议的人们阅读理解。在理解了协议内容的基础上,开发人员或采用手工编程的方法,或采用网络协议工程方法实现协议软件,并将其装入计算机网络中运行。

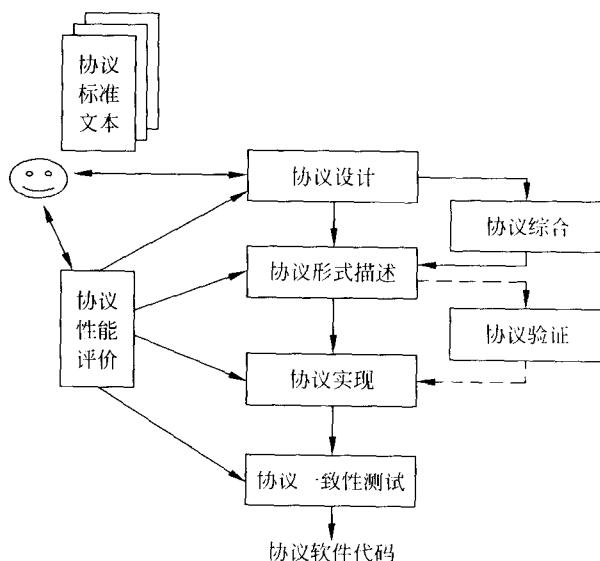
参考文献

张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999
(张尧学)

wangluo xieyi gongcheng

网络协议工程(network protocol engineering)

一门研究如何设计和构造协议规范以及如何把所设计和构造的协议规范快速、准确、低成本地转化为可执行代码的科学。它为协议的设计和开发提供一套综合的、规范的、自动的方法。协议工程包含两方面内容,一方面是协议规范(包括语法、文法和语义)的制定,另一方面是高效率、低成本的自动或半自动的协议软件开发方法的研究。网络协议开发的过程与步骤如图 1 所示。



(虚线表示未经协议综合过程时必须进行的部分)

图 1 协议开发过程与步骤

协议规范与标准的制定大多由国际标准化组织负责,例如国际电信联盟(ITU),国际标准化组织(ISO)以及美国电气与电子工程师学会(IEEE)等。而与 Internet 有关的各种协议规范与标准的制定则由 Internet 协会下属的因特网工程任务部(IETF)负责。IETF 根据 Internet 发展的技术趋势,把 Internet 相关技术划分为 10 多个领域和设置了 100 多个工作小组,这些工作小组在 Internet 上靠电子邮件互相交换信息并进行联系,对共同感兴趣的协议进行研究,然后提出相应的请求评论(RFC)提案草案或标准。协议工程的另一个内容是软件开发过程的工程化,主要研究协议的开发方法。由于协议开发仍是软件的开发,因此,软件工程的方法也可用来开发协

议软件。协议软件与一般软件的主要区别表现在:

- (1) 协议软件运行在分布式网络环境下,比较复杂,规模也较大。
- (2) 协议软件具有开放性,对标准化要求高。
- (3) 由于是分布式网络环境下运行的软件,一般具有较多的潜在错误,且难于发现和排除。
- (4) 协议软件的实现依赖于网络所提供的硬、软件平台和环境,软件移植性较差。
- (5) 安全性要求高。

协议开发必须包括以下几个方面:

- (1) 协议设计 协议设计包括概要设计、结构和模块设计、实现环境分析和详细设计、验证与测试过程设计等。协议设计的基础是协议标准规范。它

把一个用自然语言或标准形式描述语言表示的协议按照协议实现的要求从需求分析开始,逐步设计成一个符合实现平台需要的详细描述。

(2) 协议综合 协议综合从协议规范的文本出发,通过相应的综合规则和算法,抽取出协议实现所必需的功能、交互顺序和各模块所提供的服务,得到协议的形式描述,并使其不包含逻辑错误,并且具有活性与安全性等好的性质。

(3) 协议验证 通过数学方法和工具对所设计或描述的协议进行检验,证实所设计和描述的形式描述中不存在逻辑错误,即死锁、不完全性、不稳定性、活锁和信道溢出等,从而确认协议的形式描述具有安全性和活性。

使用协议综合技术设计后得到的协议形式描述不需要再进行协议验证工作,因为协议综合时所用的规则与方法已经保证了所得到的协议形式描述具备了安全性和活性。但是,由于协议综合技术很难应用于大型协议的设计过程,因而,人们仍然采用先按照协议标准文本进行形式描述,然后再将协议形式描述抽象为有限状态自动机(FSM)、Petri网(描述系统各元素的异步并发操作的工具模型)或抽象数据类型等数学模型进行验证。在验证过程中如果发现协议的形式描述中存在相应的错误,则对其进行修改后再重新验证,这一过程一直重复到验证结果中不再发现逻辑错误为止。

(4) 协议实现 协议实现把经过验证或综合设计的协议规范变换成计算机可执行代码。这是一个从形式描述到软件代码的逐步精化过程。协议实现要求首先把协议标准文本或规范抽象为面向实现的协议规范。面向实现的协议规范包括协议的应用范围、协议各层之间的服务定义、接口及功能子集的定义,与实现有关的各种参数规定,服务质量控制信息以及一致性测试条件等。协议实现方法有计算机工具辅助的半自动实现方法和完全手工编程的实现方法两种。计算机工具辅助的半自动实现方法需要有相应的形式描述语言编译器,生成的软件代码相对较长。手工编程实现的软件代码比较精练,但可能含有较多的错误。

(5) 一致性测试 (参见网络协议一致性测试)

参考文献

张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999

(张尧学)

wangluo xieyi guifan

网络协议规范 (network protocol specification) 由协议功能和服务组成的关于协议的通信顺序、逻辑、格式和行为动作的描述。网络协议规范由相应的国际标准化组织颁布。这些组织是: 国际电信联盟(ITU)、国际标准化组织(ISO)以及 Internet 工程任务部(IETF)等。这些组织接受和审查由科学家和研究小组提出的各种协议标准提案,并通过会议讨论和表决通过后成为相应的国际标准规范。

网络协议规范一般用自然语言表示,例如英语。但是,由于通信系统的复杂性和一旦发生错误后给社会或用户带来不可估量的损失,因此对于比较复杂协议的协议规范,除了用自然语言描述之外,还用形式描述语言对其进行数学的逻辑性描述。常用的协议规范形式描述语言有 SDL、Estelle、Lotos、Z 语言等。另外,有限状态自动机(FSM)、Petri 网等也常被用来描述网络协议规范。

参考文献

张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999

(张尧学)

wangluo xieyi xingshi miaoshu jishu

网络协议形式描述技术 (network protocol formal description technology) 用形式描述语言描述协议实体间信息交互规则、格式和相关定义的技术。协议形式描述主要描述三大部分,即协议实体所提供的服务、协议实体的交互行为以及该协议实体所提供的服务如何使用,也就是协议实体的用户界面。

形式描述技术的基础是基于数学的形式描述语言,例如 SDL、Estelle、Lotos 和 Z 语言等。使用这些形式描述语言对协议文本和规范进行描述,可以减少或消除由自然语言描述所带来的歧义性与不确定性。协议文本和规范中的歧义性与不确定性容易导致协议软件代码中包含无法预料的逻辑错误,并使得不同的实现人员编写出的软件代码具有不同功能,从而损害网络的互通性和互操作性。

除了上述形式描述语言之外,ASN.1 是一种经常被用来描述应用层协议的形式描述语言。例如,Internet 的简单网络管理协议(SNMP)以及管理信息库(MIB)就是用 ASN.1 描述的。形式描述语言树表组合记法(TTCN)则被用于描述一致性测试中

的协议测试集。

参考文献

张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999 (张尧学)

wangluo xieyi yizhixing ceshi

网络协议一致性测试 (network protocol conformation testing) 验证协议实现的源程序执行时所提供的功能是否与协议标准文本所规定的功能相一致的测试。

进行一致性测试的目的是为了满足网络互连的要求, 即不同的人员根据同一标准开发出来的协议软件可以通过网络进行互连和互操作。一致性测试并不检验协议标准文本中是否存在逻辑错误, 它只负责检查协议的各实现版本是否能够完成协议标准所要求的功能。

进行一致性测试需要有被测实体, 也就是被测协议软件、测试环境以及要求被测实体所完成功能的协议测试集。这些协议测试集被当作用户要求输入, 由被测实体在测试环境下运行和测试, 然后对测试结果进行分析, 如图 1 所示。

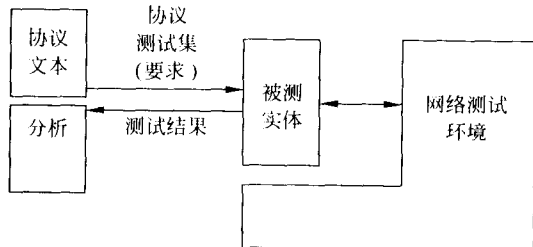


图 1 一致性测试概要

为了确认被测实体所实现的功能是否与协议标准所规定的功能相一致, 必须定义和明确协议实体所应该具有的要求和功能。这些要求和功能被称为一致性要求。一致性要求包括如下四点:

- (1) 强制性要求 在任何情况下都要进行检查的要求;
- (2) 条件性要求 在标准规定的具体条件下必须满足的要求;
- (3) 任选要求 根据具体实现不同可进行选择的要求;
- (4) 禁止要求 禁止进行检查的要求。

这些要求又可分为静态一致性要求和动态一致性要求。

静态一致性要求是为了方便网络互连而要求被测协议实体所应具有的最小能力。动态一致性要求定义被测协议实现所应具有的所有可能的行为。

一个被测实体既满足静态一致性要求又满足动态一致性要求的某个指定部分的话, 则认为该协议实现与协议标准规定的功能在指定范围内是一致的。

一般分四个方面进行协议的静态测试和动态测试, 它们是:

- (1) 基本互连测试 确定被测协议实体是否具有初步互连能力;
- (2) 能力测试 对被测协议实体进行静态一致性测试, 并与静态一致性要求相比较;
- (3) 特性测试 对被测协议实体进行动态一致性测试, 并将所得结果与动态一致性要求相比较;
- (4) 判定测试 对某个具体部分进行的小范围测试, 以进一步确认在特性测试中不能确定一致性而又可能满足某些一致性特定要求的部分。

为进行上述测试, ISO 定义了 TTCN 形式描述语言, 描述被测协议实体的静态一致性与动态一致性要求。许多测试方法和工具被开发出来用于协议的一致性测试。根据这些方法, 测试人员首先从 TTCN 描述的一致性要求中抽象测试集和选择测试方法 (因不可能对所有的一致性要求进行完整的覆盖性测试)。

测试集的选择要有代表性, 否则会漏掉协议实现一致性要求中的某些重要性质。

测试方法的选择除了测试算法之外, 还包括测试环境的选择。ISO 定义了四种基本的测试方法。即局部测试法、分布式测试法、协调测试法和远程测试法。与此对应的测试环境有三种: 局部环境、远程环境和分布式环境。这些方法分别将被测协议实体放在不同的网络环境中进行一致性测试和观察。

在局部测试法中, 被测协议实体的上下层协议实体和对等通信协议实体都处在同一主机中。被测实体与上层协议实体以及对等协议实体的交互界面可以由测试人员直接控制和观察, 如图 2 所示。

分布式测试法使被测协议可以置身于一个如图 3 所示的网络环境中。测试人员通过执行所选择的测试集, 在低层测试协议端控制和观察被测协议实体调用低层协议的服务和行为的一致性, 而在高层测试协议端控制和观察被测协议实体为高层提供服务 and 行为的一致性。所谓的低层测试协议实际上是被测协议的对等实体, 即和被测协议处于同一层

次的协议。

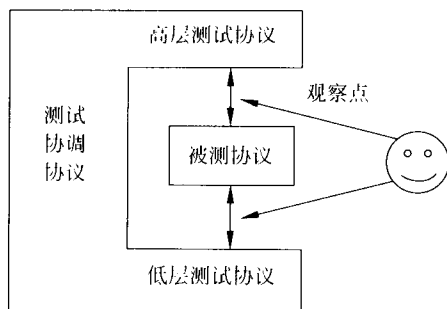


图2 局部测试法

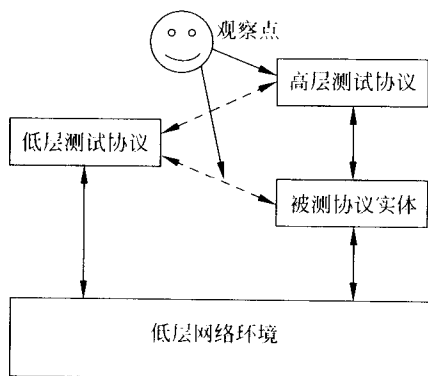


图3 分布式测试法

协调测试法是分布式测试法的增强形式。协调测试法在高层测试协议端对被测协议实体的行为进行控制和观察,这种控制和观察改在低层测试协议端进行。

远程测试法更进一步放弃了在低层测试协议端对被测协议与高层测试协议的交互行为的控制和观察。

参考文献

张尧学等. 计算机网络与 Internet 教程. 北京: 清华大学出版社, 1999

(张尧学)

wangluo yingyong fuwu

网络应用服务 (network application service) 开放系统互连基准 (参考) 模型 (OSI/RM) 中的应用层向其用户 (应用进程) 提供的一种面向应用的服务。它给应用进程访问开放系统互连环境 (OSIE) 提供相应的工具和手段。

应用层向应用进程提供的服务十分丰富, 这些

服务按不同的用途分成一些可标识的功能体, 称为应用服务要素。应用服务有公共应用服务和特定应用服务之分, 应用服务要素也随之分为公共应用服务要素 (CASE) 和特定应用服务要素 (SASE) 两大类。公共应用服务要素提供的功能与具体应用的性质无关, 在应用层中具有通用性, 如应用连接的建立和释放。特定应用服务要素提供的功能依赖于具体应用, 没有通用性。

下面给出的国际标准化组织的应用服务都是属于公共应用服务这一类的:

- (1) 虚拟终端服务 (VTS);
- (2) 作业传送和操纵 (JTM);
- (3) 文件传送、存取和管理 (FTAM);
- (4) 面向消息的正文交换系统 (MOTIS);
- (5) 图形核心系统 (GKS);
- (6) 远程数据库访问 (RDA);
- (7) 开放系统互连 (OSI) 管理。

在这些公共应用服务中, FTAM 的标准化工作进展最快, JTM 和 GKS 次之。MOTIS 与国际电报电话咨询委员会 (CCITT) 提出的 X. 400 系列建议 (报文处理系统 MHS) 实质上是相同的。对于 VTS, 由于半导体器件的飞速发展, 终端功能越来越强, 其品种也不断增加, 这给制定一个各方面都满意的虚拟终端标准带来很大的困难。至于 OSI 管理、RDA 等应用服务, 由于其自身的复杂性, 使得相应的标准化工作进展缓慢。

除了上述公共应用服务之外, ISO 在特定应用服务方面也进行了一些标准化工作, 例如银行数据处理系统的商务数据交换 (BDI) 标准和跨行业 (贸易、运输、保险、银行和海关等) 的电子数据交换 (EDI) 标准等。随着网络应用向各领域的深入开展, 各种特定应用服务也将迅速增加。 (李学农)

wangluo yunxing huanjing

网络运行环境 (network operation environment) 开放系统应用开发运行环境和网络计算环境两部分组成的集合。前者为各种应用系统的开发和运行提供了统一的支撑平台; 后者则在网络范围内将应用开发和运行环境连接在一起。

(1) 应用开发运行环境

应用开发运行环境实际上是网络中间件 (例如, 用户接口服务、网络服务、数据管理服务和数据交换服务等) 和操作系统的组合, 目前比较流行的产品有客户信息控制系统 (CICS)、通用开放系统环

境(COSE)和 Windows 开放服务结构(WOSA)。

COSA 是一个供应商联合会,其中包括 IBM、HP、SunSoft、Novell 公司。这些供应商联合推出了一个能与 Microsoft Windows 匹敌的 Unix 通用桌面系统环境(CDE)。这个组织的目标是:①为上述厂商系统所支持的通用桌面系统图形环境开发一个提供应用程序编程接口(API)的规范;②采用通用网络环境;③标识相互认可的图形、多媒体和对象技术;④定义分布式系统的系统管理。

WOSA 是 Microsoft 在 Windows 环境下为跨越不同平台的应用程序而开发的体系结构。它制定允许信息在企业内部自由流动的标准。WOSA 包括模块化应用程序编程接口(API),向任何开发人员开发的应用程序提供网络服务,这些网络服务可以包括:电子邮件、数据库和主机连接等。

WOSA 还提供插入任何开发人员开发的后端服务。从这个意义上讲,WOSA 实行的就是通常所称的“中间件”策略。它直接在操作系统中建立,并刺激可在网络上进行协作的应用程序的增长。此外,WOSA 还提供公用数据访问和安全性增强附件。

(2) 网络计算环境

网络计算环境主要实现网络范围内的数据管理、通信服务和网络管理。在数据管理方面有用于数据库间通信的远程数据访问(RDA)。通信服务方面有开放系统互连(OSI)、远程过程调用(RPC)以及分布计算环境 DCE、CORBA(参见分布式计算环境)等。网络管理方面分布管理环境(DME)和简单网络管理协议(SNMP)、公共管理信息协议(CMIP)等。

参考文献

1. Marciniak J J. Encyclopedia of Software Engineering. New York: John Wiley & Sons Inc., 1994

2. 胡道元. INTRANET 网络技术及应用. 北京:清华大学出版社,1998 (王晓春)

wangluo zhongjianjian

网络中间件(network middleware) 对应用程序屏蔽了实际网络和通信协议细节的一类支撑软件。它提供高级编程接口,帮助开发人员在不同的环境创建应用程序,而不需对使用的网络的通信协议有更多的了解。广义说来,网络中间件是在网络环境中,建立在具有基本通信协议的操作系统之上,支持应用的有效开发、部署、运行和管理的一层支撑

软件。

通常中间件在使用不同网络通信协议的客户端-服务器环境下使用。它可以对客户-服务器使用屏蔽协议,从而使开发人员集中精力于改进应用程序,而不是开发通信接口。

例如,Microsoft 的开放式数据库连接(ODBC)标准提供后台数据库系统操作的通用功能,可通过将前台应用程序写入 ODBC 来利用这些功能。ODBC 屏蔽了不同厂商结构查询语言(SQL)实现的区别。Microsoft 以一组 Microsoft Windows 驱动程序的形式提供 ODBC,以提供对 Microsoft Access、Microsoft Excel、Microsoft SQL Server、FoxPro、dBASE、IBM DB2 和 Oracle 等格式产生的数据的访问。ODBC 是为了使 Windows 成为客户访问后台数据库的标准而设计的。

中间件的应用场合包括:①用户需要访问许多不同的后台服务器上的服务。②后台服务器可以使用不同的操作系统,并采用不同的通信协议。③后台数据库服务器有不兼容的 SQL 命令集,使用户和程序员很难从一个系统转到另一个系统。④用户需从不同的应用程序来访问服务,而不是特定的应用程序。⑤用户需在多个厂商环境下使用多种协议在任何前台访问任何后台服务。

要在多协议、多厂商产品的环境下使用中间件,程序员只需简单地编写到中间件的接口即可,而由中间件来处理由多协议、多厂商产品所带来的问题。

(胡道元)

wangluo ziyuan yuyue xieyi

网络资源预约协议(network resource reservation protocol, RSVP) 用于 Internet 中点到点通信和点到多点通信的对网络资源进行预约的协议。

RSVP 具有下述特点:

(1) 可在 Internet 上以点到点方式或点到多点方式进行资源预约;

(2) 预约资源时,采用单方向预约方式,即由数据流的接收端向数据源端沿路径进行预约。这种方式有利于客户-服务器模式的通信方式以及资源的合并与共享。在图 1 中,RSVP 由数据接收端 D_i 向数据发送端 S_i 沿路径进行资源预约,且在路由器 R_1 和 R_2 之间共享带宽;

(3) 在路由器等网络元素上设置,维护记录路由和资源预约信息的软状态表,并能根据路由和预

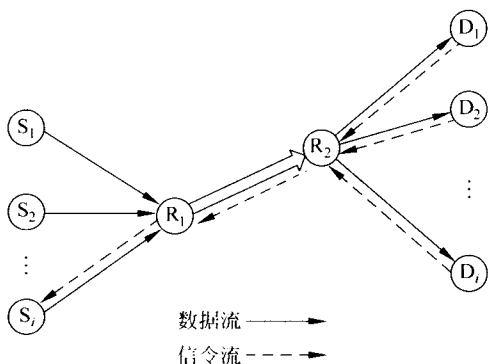


图1 数据流与预约信令流示意图

约信息的变化进行自动更新和调整;

(4) 能根据用户对数据源的访问需要提供不同的预约方式;

(5) 提供流量控制和传输策略控制;

(6) 既支持 IPv4 协议,也支持 IPv6 协议等;

(7) RSVP 是控制信令协议,主要由控制报文、控制报文参数、控制状态和预约风格组成。

RSVP 的控制报文有两类:控制 (PATH) 类报文和预约 (RESV) 类报文。PATH 类报文由数据源端发出,RESV 类报文则由数据接收端作为对 PATH 类报文路径中各网络元素的资源要求沿 PATH 类报文设置的路径返回发出。如果接收端不需要预约资源,则不返回预约类报文,而直接沿相应的路径接收来自于源端的信息。RSVP 协议中的路径与消息如图 2 所示。

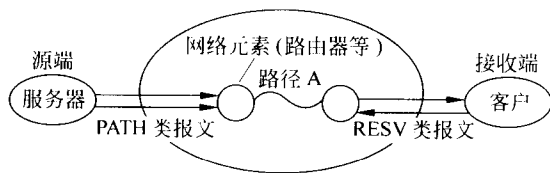


图2 RSVP 协议中的路径与消息

是否发送 PATH 类报文或 RESV 类报文,由 RSVP 的会话功能决定。一个 RSVP 会话定义一个预约需要的点到点传输或点到多点传输的数据流。RSVP 的会话由相应的目的地址、传输层协议和会话端口来定义与识别。RSVP 可在 IP 协议上实现,也可在用户数据报协议 (UDP) 上实现 (参见 TCP/IP 协议集)。

RSVP 通过相应的 PATH 类报文和 RESV 类报

文沿着数据流路径的所有结点传递服务质量 (QoS) 的控制参数和要求,并在路由器等网络元素上建立和维护 QoS 控制服务用的路径状态和预约状态,这些状态被称为软状态。软状态实际上是一种无连接、定期刷新储存在网络元素相应表格中的路径状态和预约状态信息的工作方式。

针对不同的资源预约需要,RSVP 定义了不同的资源预约风格与模式。这几种模式是:固定模式、通配符模式和显式共享模式。其中,固定模式是固定的接收端对固定源端的预约;通配符模式是指所有预约同一源端的不同接收端用户可以共享同一路径中的带宽和缓冲区等资源;使用显式共享模式的不同接收端用户在共享路径中的带宽与缓冲区等资源的同时,还可以选择指定不同的信息源。

参考文献

张尧学等. 计算机网络与 Internet 教程. 北京:清华大学出版社,1999 (张尧学)

wangqiao

网桥 (bridge) 位于开放系统互连基准 (参考模型 (ISO/RM) 的第二层——数据链路层的一种网络互连设备 (见图 1)。它具有差错检验、寻址、必要的地址和数据格式修改等功能,且采用存储转发方式工作。

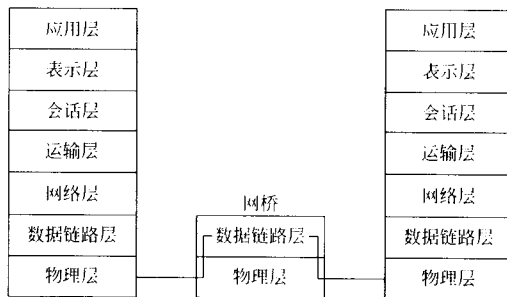


图1 网桥在数据链路层上互连网络

网桥往往用于连接两个或多个局域网。利用网桥的筛选功能,可以适当地隔离不需要传播的信息,从而改善网络性能,提高整个网络的吞吐量和响应速度。例如,某大学有数千台计算机,并配有若干台服务器。如此多的计算机如果连成单个局域网,则对带宽的要求非常高。如果把它分割成若干个局域网,每个局域网有自己的文件服务器,然后用网桥把这些局域网互连起来。这样,大部分的通信被限制在单个局域网之内,因而对单个局域网的带宽要求

就大幅度降低了。此外,用网桥来互连局域网,还可以防止某些结点的故障波及到整个网络。而且采用网桥还可以限制 sniffer(一种局域网黑客软件)的攻击,对于提高整个网络的安全性和稳定性有相当大的作用。

为了使不同类型的局域网能够互连,可以使用透明网桥、源路由网桥、源路由透明网桥、源路由翻译网桥,等等。

对于现在的局域网来说,网桥的使用越来越少。如果有需要,一般把路由器进行专门的配置作为网桥来使用,或直接使用智能性网桥组合——交换机。交换机是由网桥发展而来,由于端口数量多,网络延时小,正逐步取代传统网桥的地位。(张世永)

wangzhuang shujuku

网状数据库 (network database) 采用网状模型的数据。

网状模型用网状结构表示各类实体及其间的联系。在网状结构中:

- (1) 允许一个以上的结点没有双亲;
- (2) 一个结点可以有多个的双亲。

网状模型中每个结点表示一个记录类型(简称记录型),每个记录类型可包含若干个字段,结点间的连线表示记录型间的联系。记录型描述实体,字段描述实体的属性。网状数据库中记录型之间的联系不惟一。因此,要为每个联系命名,并指出与该联系有关的双亲记录和子女记录。

图1是一个网状模型的例子:

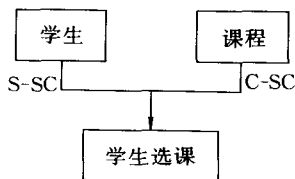


图1 一个简单的网状模型

在此例中,一个学生可以选修若干门课程,某一门课程可以被多个学生选修,学生与课程之间是多对多的联系。引入学生选课的联系记录,从而把学生与课程之间是多对多的联系分解为学生与学生选课,课程与学生选课之间两个一对多的联系。学生与学生选课之间的联系被命名为S-SC,课程与学生选课之间的联系被命名为C-SC,学生和课程为学生选课的双亲结点。

网状数据库的典型代表是DBTG系统,亦称CODASYL系统。这是20世纪70年代数据系统语言研究会CODASYL下属的数据库任务组DBTG提出的一个系统方案。虽然它仅是一个方案,但它所提出的基本概念、方法和技术具有普遍意义。很多实际运行的网状数据库系统,如IDMS、IDS/2、DMS1100等,都是以DBTG模型为基础,通过对其进行简化和修改而形成的。

DBTG的系统结构由子模式、模式、内模式组成,它是一个典型的三级模式结构。

DBTG是宿主语言系统。用户用宿主语言(如COBOL)编写应用程序,并在需要访问数据库时嵌入数据操纵语句完成对数据库的操作。DBTG网状数据库中各记录型之间的联系用系来描述。数据库中既要储存数据本身,还要反映出数据之间的联系。网状模型中,系通常以链的方式来实现,包括单向链、双向链、环状链、非环状链、向首链等,对系的每一个系值从首记录开始按系序用指引元依次链接各属记录。

网状模型是一种比层次模型更具普遍性的结构,它去掉了层次模型的限制,允许多个结点没有双亲结点,允许结点有多个双亲结点,此外它还允许两个结点之间有多种联系(称之为复合联系)。因此网状数据模型可以更直接地去描述现实世界。而层次结构实际上是网状结构的一个特例。

网状数据库虽解决了层次数据库存在的一些问题,但是,在网状数据库中,用户对数据库的存取必须沿着存取路径到达目标数据,这就必须随时记录数据库在各个范围中的当前值,加重了用户的负担;对数据的操作是一次一个记录的存取方式,程序和数据虽有较高的物理独立性,但逻辑独立性不高。

网状数据库系统在20世纪70年代与80年代初非常流行,在数据库系统产品中占主导地位,虽然近年来逐渐被关系数据库系统取代,但目前在美国、加拿大等国家,由于历史原因,网状数据库的用户数仍然很多。

参考文献

1. 萨师煊,王珊. 数据库系统概论(第二版). 北京: 高等教育出版社, 1991
2. Date C J. An Introduction to Database System. 6th Edition. Reading: Addison Wesley Publishing Company, 1995

(王珊)

weibo tongxin

微波通信 (microwave communication) 不使用固体传输介质,利用微波在两点间传送信号的一种通信。在两点间直线距离内无障碍时就可使用微波传送,例如卫星到地面、城市两个建筑物之间、无法布设电缆的开阔区域(如沙漠、草原和湖泽)等。

微波传输系统使用双向天线,以点到点方式从一点向其他点发射并接收其他点发出的电磁波或无线电波的能量。这些天线需要无障碍的路径,在地面上最大范围可达 30 千米。天线通常安装在高塔上,以扩展它们的工作距离,并避开引起反射信号的障碍物。

与无线电广播不同,微波通信是一个点到点的通信系统,使用配备了信号放大器的中继站以扩展微波通信的范围。一个中继站一般有两个天线,分别用于指向远方不同的方向,传输频率为 1 ~ 30 GHz。

安装一个小型微波系统是很简单的。例如,可以在两个建筑物上分别安装一个天线,并将这两个天线指向相对,这样就建立了两个建筑物之间的微波通信。由于不需租用电信公司的传输信道,因此建造系统的费用低,且比铺设电缆更加方便。

参考文献

Sheldon T. LAN Times Encyclopedia of Networking. McGraw - Hill Inc., 1994 (胡道元)

weichengxu kongzhiqi

微程序控制器 (micro-programmed control unit, MCU) 通过执行由若干条比机器指令低一层次的微指令所组成的微程序而实现机器指令所必需的各种基本操作的控制器。

微程序设计思想最早是英国的 M. V. Wilkes 在 1951 年首先提出的。但是在相当长时间内,由于缺乏快速的微程序存储器而未能推广使用。随着高速半导体只读存储器的发展,1964 年 IBM 360 系列计算机成功地采用了微程序控制方案,方便地实现了不同计算机间指令兼容问题,从此微程序控制方案获得广泛应用。

在采用微程序控制器的计算机中,每一条机器指令的执行可以分解为一系列更为基本的操作,称为**微操作**。控制进行各种微操作的信号称为**微命令**。根据指令功能的分步要求,将可同时执行的微命令组合在一起,形成**微指令**。与每一条机器指令

相对应的一段**微程序**是用微指令编写的一段微指令序列。在控制器内部存放微程序的存储器称为**控制存储器**或**微存储器**,它一般由只读存储器芯片 ROM 或可编程只读存储器芯片 PROM 组成。

微指令分类

微指令可分成水平型微指令和垂直型微指令两大类。

水平型微指令 由控制字段和下址字段构成如下:

控制字段	下址字段
------	------

控制字段中的每一个二进制码位代表一个微命令,用来控制相应的微操作,如某一位指定为“加法微命令”,则当该位为 1 时,控制 ALU 进行加法运算;为 0 时,ALU 不进行加法运算。下址字段用来决定下一条即将执行的微指令的地址(称为后继微指令地址)。由于复杂指令集计算机的微命令很多,因此水平型微指令的控制字段的长度一般在百余位到几百位之间。这种微指令操作并行度高,编写的微程序短,速度快,使用灵活,适用于大中型高速计算机。缺点是微指令字较长,控存位码利用率不高,形成立即数和转移地址困难。

垂直型微指令 其格式与机器指令很相似,通常一条微指令完成一种基本运算或基本操作,如加法微指令、移位微指令、转移微指令等。微指令中设置微操作码, N 位微操作码可表示 2^N 种微指令。微指令中还设置源寄存器地址字段和目的寄存器地址字段,指明操作数的地址。微程序通常是顺序执行的,微程序控制器中还设置微指令计数器。为提高操作并行度,微指令字中还可增加辅助功能字段。基于垂直型微指令的微程序称为垂直型微程序。其特点是:①微指令字较短,控存位码利用率高;②编制微程序容易;③可以采用高级微程序设计语言,容易实现设计自动化。其缺点是操作并行度低,编写的微程序较长,完成一条机器指令时间较长。

分段编码法水平型微指令 为了兼顾速度指标与经济性,在满足速度要求的前提下尽量缩短微指令字长,通常采用分段编码法。其原理是将水平型微指令的操作控制字段按照数据通路及微操作的互斥性分成若干小组(称为字段)。每组微命令用若干位二进制编码表示,通过译码器控制相应微操作的执行。各个控制字段译出的微命令可以同时出现,实现并行操作。

毫微程序 通常采用二级控制存储器结构,第

一级控制存储器存放垂直型微指令,用以解释指令;第二级控制存储器存放水平型微指令,用以解释垂直型微指令,执行指令过程中由垂直型微指令调用水平型微指令。采用上述结构的微程序称为毫微程序。

微程序的顺序控制

根据现行微指令得出下条微指令在控制存储器中的地址,即后继微指令地址,有以下两种情况:

无分支流程 微指令的后继地址是惟一的。可以设置微程序计数器 μPC , 每执行一条微指令, μPC 加 1, 形成下一条微指令地址, 这是顺序执行的情况。也可以由微指令的下址字段指定下一条微指令地址, 实现顺序执行或无条件转移。

分支流程 一条微指令完成后, 根据测试条件决定转向哪一条后继微指令。

实现分支流程(条件转移)的方法有多种, 如:

(1) 用类似机器指令条件转移的方法, 产生下一条微指令地址。

(2) 在下址字段中, 设置若干位较短的修改地址字段, 用来指出转移地址的低位部分, 形成不同的分支地址。

(3) 用测试条件去修改微指令后继微地址字段, 形成不同的分支地址。

(4) 用可编程逻辑阵列 PLA 给出不同条件下的分支地址。

微程序设计的方法与程序设计类似, 可以设计成循环微程序、微子程序与公用微程序等。其中微子程序与公用微程序指可被多个微程序调用的一段微程序。微子程序保存返回微地址, 微子程序最后一条微指令应具有返回功能。

微程序控制器的组成

微程序控制器逻辑框图举例如图 1 所示。除了控制器常设的一般部件, 包括程序计数器 PC、指令寄存器 IR、时序电路和中断逻辑以外, 还有保存微程序的控制存储器 CM 以及为执行微程序而设置的各种电路。简述如下:

控存地址寄存器 CMAR 保存即将执行的微指令在控存中的地址。

微指令寄存器 μIR 包括操作控制字段 DOCF, 转移控制字段 BCF 和转移地址字段 BAF。DOCF 产生各种微命令, BCF 给出转移条件, BAF 提供转移地址。通过条件测试电路决定后继微地址, 送控存地址寄存器。

机器指令的微程序入口地址由机器指令操作码

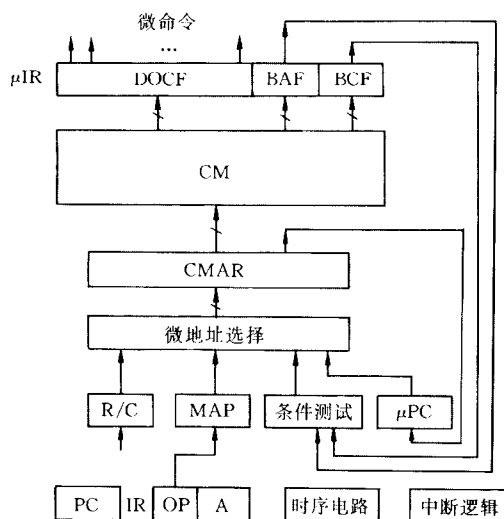


图 1 微程序控制器逻辑框图

通过地址映射电路 MAP 得到; 当微指令顺序执行时, 可通过 μPC 加 1; 当执行转移时, 可通过条件测试电路决定后继微地址; 当执行转移微子程序时, 返回微地址可保存在 R/C (返回-计数) 寄存器中。当执行循环微程序时, 在 R/C 中保存循环次数, R/C 有计数功能。

参考文献

1. 金兰. 计算机组织与结构. 北京: 高等教育出版社, 1986
2. 陈炳从. 电子计算机微程序设计技术. 北京: 国防工业出版社, 1981

(谢树煌)

weichuliqu

微处理器 (microprocessor) 具有中央处理器功能的大规模集成电路器件。它是微型计算机的核心部件。

微型计算机的传统结构 如图 1 所示, 其中虚线框内是微处理器, 它包括 3 个基本部件。

(1) 运算器 它既能执行算术运算, 又能执行逻辑操作。

(2) 寄存器 每个微处理器中都有多个寄存器, 用来存放操作数、中间结果以及标志工作状态的信息等。

(3) 控制器 它包括控制操作的电路以及用于定时的时钟脉冲发生器等。

自从 1971 年第一个 4 位微处理器芯片 Intel 4004 问世以后, 很快就有大量 8 位微处理器如 Intel

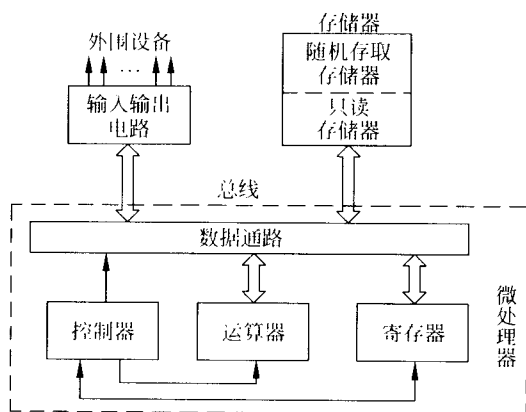


图1 微处理器与微型计算机的结构框图

公司的 8080、Motorola 公司的 6800、Zilog 公司的 Z80 等推向市场。在经历了 16 位微处理器发展阶段后，32 位或 64 位微处理器风靡全球。Intel 公司继 1987 年推出 32 位的 80386 微处理器后，不久又推出 80486 微处理器，其速度达到 54 MIPS。1993 年，Intel 公司推出的 Pentium 微处理器，采用 0.8 μm 的 BiCMOS 工艺，集成了 310 万只晶体管，在时钟频率为 66 MHz 时，功耗为 13 W。1994 年，Intel 公司推出时钟频率为 100 MHz 的 Pentium 微处理器。1995 年又推出时钟频率为 150 MHz 和 166 MHz 的 Pentium Pro 微处理器。以后，Intel 把 Pentium Pro 改称为 Pentium II，并推出 Pentium III 和 Pentium 4。目前其微电子工艺已做到 0.13 μm ，主频可达 3 GHz 以上，片上晶体管数达 4 000 万个。Intel 把这一系列 32 位字长的微处理机称为 IA32 体系结构。2000 年，Intel 与 HP 合作，又联合开发研制出 64 位的微处理机 Itanium，称为 IA-64 体系结构系列，主要用于服务器产品市场。AMD 公司的产品与 Intel 产品相兼容，并且与 Intel 公司进行了竞争。

1992 年，DEC 公司率先推出 64 位 Alpha 芯片。时钟频率为 133 ~ 200 MHz 的 Alpha 21064 微处理器采用了 0.75 μm 的 CMOS 工艺，在 13.9 mm × 16.8 mm 的硅片上集成了 168 万只晶体管。

现代微处理机与 80 年代的 80X86 微处理机有很大的不同。主要方面有：①从每个周期发射一条指令，发展到发射多条指令；使多条指令尽可能地并发执行，成为微处理机设计考虑的重要问题。②指令的执行多采用流水线操作。③微处理机都带有片上的高速缓存，包括指令高速缓存和数据高速缓存，以及包括第二级高级缓存。这些高速缓存在片上的

晶体管数，往往占有片上晶体管总数的 50% ~ 60% 以上。现代微处理机已和传统的微处理机有很大的不同（参见处理机体系结构）。

微处理机的性能每 18 个月增长一倍，这就是著名的摩尔定律。微处理器的发展速度如此迅速，其原因主要有以下 3 点：①随着半导体微电子技术的发展，器件本身的特性大幅度提高；②出现了以 RISC（精简指令集计算机）为代表的微处理器体系结构；③超大规模集成电路（VLSI）器件的 CAD 技术使设计质量和效率大为提高。

参考文献

1. 李三立. 微处理器与微型计算机. 北京：国防工业出版社，1981
2. 李三立. RISC——单发射与多发射体系结构. 北京：清华大学出版社，1994 （李三立）

weikongzhiqi

微控制器（micro-controller） 由中央处理器（CPU）、存储器（包括 RAM 与 ROM）、输入输出（I/O）端口（包括并行 I/O、串行 I/O、模数转换器）、计时器和计数器等组成，具有完整数字处理功能的大规模集成电路。它是一种面向 I/O，适合于控制领域应用的集成计算机芯片，目前主要用于工业控制、数据处理、信号处理、智能仪器、通信产品及民用消费产品。通常也把它简称为 MCU。MCU 配以适当的外围设备和软件就可构成一个计算机应用系统，所以也称之为单片微型计算机，简称为单片机。

MCU 的发展始于 20 世纪 70 年代中期，其间 MCU 称为单片机。由于工艺和集成度的限制，一个完整功能的 MCU 由两块集成电路组成，如 Fairchild 公司的单片机 F8 必须外接一块专为 F8 设计的程序存储单元电路 3851。第二阶段为低性能 MCU 阶段，虽已只用一块芯片构成，但性能低，品种少。如 Intel 的 MC 8-48 系列，芯片内含有 CPU、并行 I/O 口、计时器、RAM 和 ROM 等，但其 CPU 功能不强，I/O 的种类和数量少，存储容量小，只能应用于要求比较简单的场合。第三阶段始于 70 年代末，是高性能微型计算机系统（MCS）发展阶段。MCU 内部具有功能很强的 CPU、比较多的输入输出电路和大容量的数据存储器、程序存储器。MCU 产品型号、规格多，各具特色，能满足不同领域应用需求。

MCS 分为通用型和专用型两大类。通用型 MCS 是把可开发资源（如 ROM、I/O 口等）全部提供

给用户,以下所讨论的 MCS 单指通用型 MCS。专用型 MCS 实际上是一种微控制系统集成化的产品,如频率合成调谐器、打印机控制器等。按 MCS 基本操作处理的数据位数区分,可有 1 位 MCS、4 位 MCS、8 位 MCS、16 位 MCS 和 32 位 MCS 等。

随着大规模集成电路技术的发展,微控制器的集成度也在不断提高。90 年代初期微控制器的集成度已超过 100 万个等效晶体管,到 2000 年可望超过 5 000 万个等效晶体管。作为一个自成体系的单片微型计算机,除了 CPU 之外,微控制器还包括超高速缓冲存储器、主存储器、输入输出接口、直接存储器存取(DMA)控制器、中断处理器、计时器以及其他高性能微控制器所必需的子系统。

图 1 为美国 Motorola 公司的 8 位微控制器 68 HC11 的结构框图。由图可见,68 HC11 有 5 个通

用并行端口,其中端口 C、D 的引脚可逐一配置作为输入输出之用,其余 3 个端口在结构中是固定的,A 为混合端口,B 为输出端口,E 为输入端口。68 HC11 有两根联络信号线,可以使端口 C 设置为联络信号式输入端口或联络信号式输出端口。它还有一个逐次逼近模数转换器和一个模拟多路转换器,可将端口 E 的一个或多个输入引脚的电压转换成数字信号。当出现按钮复位、上电复位、电闪复位和监视计时器复位这四种情况之一时,复位电路执行复位功能。微控制器的中断电路和可编程计时器提供与外围设备进行实时交互的能力。I/O 串行端口用于扩展输入输出线并驱动具有移位寄存器的输入输出设备。通用异步收发(UART)串行端口提供与带有 UART 串行端口的设备连接的标准接口。

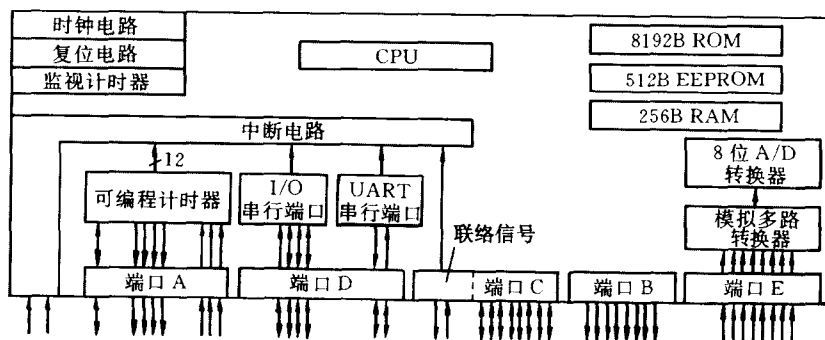


图 1 Motorola 68 HC11 结构框图

68 HC11 有一些指令可以处理 16 位的操作数,在数据总线上传送一个 16 位的操作数要占用 2 个周期。68 HC11 有 16 位的内部地址总线,可以访问 2^{16} 个地址。它有 3 种存储器:只读存储器(ROM)用作程序存储器,随机存取存储器(RAM)用作数据存储器,电可擦编程只读存储器(EEPROM)用作非易失性数据存储器(存储器中的数据不因断电而丢失)。

微控制器正朝着高性能和多品种的方向发展。MCU 性能的提高主要体现在下述 4 个方面:①增强 MCU 内部 CPU 功能,即提高其数据处理速度和精度。包括扩大字长,增加乘法、除法部件,采用流水线结构等。有的 MCU 则将高性能 16 位或 32 位微处理器原封不动地移来,作为 MCU 的 CPU 单元,这样 MCU 便与相应的通用计算机系统的软件兼容,具有相同的数据处理能力,便于开发和应用。②增加 MCU 内部资源的集成,尽量减少外接电路,使 MCU

本身即是一个完整的应用系统。例如增加内部存储器种类及容量,扩大 I/O 端口品种和数量。I/O 端口可以有串行口、并行口、多路模数转换器、计时器、定时输出和捕获输入、系统故障监视器、DMA 通道及数模输出电路等。③设计和使用多功能复用管脚,使 MCU 内部资源的增加不会导致 MCU 外引脚的过多增加,从而提高应用的灵活性。④扩大外寻址范围以提高系统的扩展功能。目前外寻址空间已从典型的 64 k 字节扩大到几兆字节。MCU 品种的发展趋势是多品种和多层次,主要体现为:①发展低电压和低功耗品种,如电池供电系统和野外作业系统。②发展微型化 MCU,即利用 MCU 设计模块化的结构特点,在内核 CPU 不变的情况下根据应用目标调整内部各功能模块的规格和外引脚数,以发展新品种。

目前,一些微控制器的性能已可与主流微处理器相媲美,其应用已经渗透到工业和管理的各个方

面。在机器人、机械工具、汽车、飞机、宇宙飞船、医疗电子设备等许多仪器和装置上,微控制器都起着重要的作用。

参考文献

1. Hintz K J, Tabak D. Microcontrollers: Architecture, Implementation, and Programming. New York: McGraw-Hill Inc., 1992
2. Peatman J B. Design with Microcontrollers. New York: McGraw - Hill Inc., 1988 (马玉海)

weineihe

微内核 (microkernel) 操作系统中仅包含为所有应用所必需的资源控制与通信功能的内核。内核是操作系统中最靠近硬件且享有最高特权的一层。微内核功能以外的功能作为服务程序在内核外实现,供使用时调用,这样,可使操作系统之内核为最小。

20 世纪 70 年代,随着操作系统的功能日益增强,系统日益复杂庞大,其内核亦随之增大,从而出现了微内核概念。其优点是使操作系统易于理解、实现、维护和移植,系统服务剪裁与配置较为灵活,利于适应不同应用的要求,但是,置于内核外的系统服务效率将会下降,含于内核内的功能的灵活性受到限制。因此,在设计具微内核结构的操作系统时重要的考虑因素是性能与灵活性之间的权衡以及易维护性与系统开销之间的权衡。

为使在不具微内核结构的操作系统上运行的应用程序也能在具微内核结构的操作系统上继续使用,通常在具微内核的操作系统与应用程序之间提供一个针对不具微内核结构的操作系统(如 UNIX)的仿真接口,将应用程序的系统调用转换成对具微内核结构的操作系统的调用。

美国卡内基-梅隆大学开发的 Mach 是一个著名的具微内核结构的操作系统。Mach 与 UNIX BSD 完全兼容。但 Mach 3.0 采用一个很小的内核,将 UNIX 操作系统功能都移到内核以外,并实现了多种仿真接口。例如,Apple MacOS X 服务器操作系统也采用了 Mac 内核。

Microsoft Windows NT (参见 Windows 操作系统)可以看作一个混用层次结构与微内核结构的例子。Windows NT 可以运行 Win32、OS/2、POSIX 等应用,其内核支持客户应用与应用服务器之间的消息传递,而在内核以外运行针对各种应用类型的服务器。

参考文献

1. Bacon J. Concurrent Systems—Operating Systems, Database and Distributed Systems: An Integrated Approach (2nd edition). Addison-Weisley, 1998
2. Silberachatz A, Galvin P B, Gagne G. Operating System Concepts (6th edition). John Wiley & Sons, Inc., 2002 (陈道蓄)

weixing jisuanji

微型计算机 (microcomputer) 以微处理器为中央处理器而组成的计算机系统。又称微型机或微机。早期的微型计算机由微处理器芯片、半导体存储器及其他辅助电路安装在印刷电路板上,再配置必要的外围设备组成。有的微型计算机只有一块电路板,称为单板微型计算机,或简称单板机。随着微电子技术的发展,微型计算机电路可以在一个芯片上,称为单片计算机,或简称单片机。

发展简史

微型计算机的历史是从 1971 年美国 Intel 公司推出 4004 微处理器开始的。按处理数据的通路宽度和功能,微型计算机的发展大致可以分为 5 个阶段。

第一阶段(1971 年—1973 年):典型的微型计算机以 Intel 公司的 4 位微处理器 4004 和 4040 为基础。微处理器和存储器采用 P 沟道金属氧化物半导体(PMOS)工艺,工作速度很慢。微处理器的指令系统不完整。存储器的容量很小,只有几百字节。微型计算机中没有操作系统,只有汇编语言。这种微型计算机主要用于工业仪表、过程控制或计算器中。

第二阶段(1974 年—1977 年):代表性的微型计算机以 8 位微处理器为基础。典型的微处理器为 Intel 公司的 8080 和 8085、Zilog 公司的 Z 80 及 Motorola 公司的 6800。微处理器采用 N 沟道金属氧化物半导体(NMOS)工艺,具有较完整的指令系统和较强的功能。存储器容量达 64 KB,配有荧光屏显示器、键盘和软磁盘等输入输出设备,构成了独立的台式计算机。在微型计算机中配备有简单的磁盘操作系统(如 CP/M)和高级语言。这种台式微型计算机又称为个人计算机。实际上,凡是最初设计来为单个用户独自使用其计算能力的,且价格低廉得使个人能够负担的微型计算机,都称为个人计算机。

第三阶段(1978 年—1981 年):代表性的微型计算机以 16 位和准 32 位微处理器为基础。典型的

微处理器有 Intel 公司的 8086, Motorola 公司的 68000 和 Zilog 的 Z8000。微处理器采用短沟道高性能 NMOS 工艺。在体系结构方面,吸收了传统的小型计算机甚至大型计算机的设计思想,如虚拟存储和存储保护等。这时的微型计算机已有相当强的功能,存储容量可达 1 MB,还可配备较大容量的软磁盘和硬磁盘。在这一阶段,操作系统、高级语言、工具软件和应用软件也日益成熟、丰富。在此期间,多用户微型计算机系统、多处理机微型计算机系统已开始出现。工业控制微型计算机等也得到了发展。

第四阶段(20 世纪 80 年代初期至中期):80 年代初,IBM 公司推出开放式的 IBM PC 是微型计算机发展的一个里程碑。IBM PC 采用了 Intel 80x86 (当时为 8086,80286 和 80386)微处理器和 Microsoft 公司的 MS-DOS 操作系统,IBM 公司还公布了 IBM PC 的总线设计。这 3 个方面的开放,为微型计算机的大规模生产打下了基础。很多公司纷纷研制与 IBM PC 兼容的微型计算机及其配套的板级产品和外围设备,很多软件公司研制和开发在 MS-DOS 基础上的软件。当时,IBM PC 所用的芯片、操作系统和总线实际上形成了国际性的工业生产的主要标准,从而使微型计算机的生产发展为规模经济的产业,推动了微型计算机应用的飞速发展。与此同时,美国 Apple 公司推出的微型计算机具有菜单式的选择功能和图形用户界面,使微型计算机的使用更加方便。

第五阶段(从 20 世纪 80 年代中后期开始):RISC(参见**精简指令集计算机**)技术的问世使微处理器的体系结构发生了重大的变革。RISC 微处理器的设计周期短,工作速度快。在 1987 年 RISC 微型计算机进入批量生产时,其运算速度已达每秒几千万次,后来的 RISC 微型计算机可达每秒几亿次,并且可以采用 UNIX 操作系统。这种变革使微型计算机、小型计算机和大型计算机的界限越来越模糊,并且可以做到使它们的软件二进制兼容。

分 类

微型计算机分类的方法很多。从处理数据宽度来分类,可分为 8 位微型计算机、16 位微型计算机、32 位微型计算机和 64 位微型计算机。

从组装形式来分类,微型计算机可以分为便携式微型计算机和非便携式微型计算机。非便携式微型计算机又可分为台式计算机和塔式计算机;便携式微型计算机是一种可移动的计算机(参见**移动式计算机**)。有的便携式微型计算机有较大的显示

屏,连同磁盘和键盘可全部装入一个手提箱中,称为手提式计算机。早期的便携式微型计算机分为膝上计算机、笔记本计算机和掌上计算机。后来膝上计算机趋于淘汰,便携式微型计算机的体积向更小的方向发展,而且输入方式更加多样化,如笔输入。后来把便携式计算机分为笔记本计算机、手持计算机和笔输入计算机。

根据微型计算机是否由最终用户使用,微型计算机可分为独立式微型计算机和嵌入式微型计算机(参见**嵌入式计算机**)。独立式微型计算机由最终用户直接使用,最常见的是个人计算机。嵌入式微型计算机作为一个信息处理部件装入一个应用设备中,最终用户不直接使用计算机,使用的是该应用设备,例如包含有微型计算机的医疗设备、高级录像机等。嵌入式微型计算机一般是单片机或单板机。

根据用途来分类,微型计算机可分为通用微型计算机和专用微型计算机;也可分为民用微型计算机、工业用微型计算机和军用微型计算机。工业用微型计算机和军用微型计算机对于环境适应能力、抗干扰能力等的要求比民用微型计算机的高,因而采用加固的组装结构,而且往往要求有中断响应快的实时操作系统。

硬件结构

微处理器是微型计算机的核心部件,它的基本组成部分包括:运算部件、寄存器组、控制部件以及由数据线、地址线和控制信号线组成的内部总线。微处理器能够完成取指令、指令译码、取操作数、执行指令、送结果等操作,以及按状态字执行特定操作,例如处理异常事件、执行与外围设备交换信息的操作等。

微型计算机的存储部件包括随机存取存储器、只读存储器以及相应的读写电路和控制电路。高档微型计算机包含高速缓冲存储器。新型的微型计算机还包含存储管理部件 MMU,它主要用于分配存储空间,并保护某些特定的存储空间,防止不合法的访问。存储管理部件可以与微处理器集成在同一芯片上,也可以单独做成一个芯片。

一般的微型计算机包括输入输出接口和常规的外围设备,即键盘、鼠标器、荧光屏显示器、打印机、软磁盘和硬磁盘。单板机的输入设备可以是小键盘,输出设备可以是发光管或液晶显示屏。仪器或工业控制用的微型计算机一般需要模数和数模转换设备。新型的微型计算机还可配备多种通信设备、语音、图像或手写体文字的输入输出设备等。

发展趋势

应用最广、产量最高的两种微型计算机是个人计算机和单片机。前者年产量为几千万台,后者为几亿片。单片机已经广泛用于家电、生活用具和仪器仪表,正在向智能化发展。个人计算机除了处理数据、文字外,还可成为处理图形、图像、语音和声音的多媒体计算机。个人计算机和通信更紧密地结合并作为网络的终端机是微型计算机的一个重要发展趋势。使多台微型计算机并行工作,可以实现性能价格比高的高性能计算机系统。

参考文献

1. 李三立. 微处理器与微型计算机. 北京: 国防工业出版社, 1981
2. 张福炎等. 微型计算机 IBM PC 的原理与应用. 南京: 南京大学出版社, 1984 (李三立)

weizuzhuang jishu

微组装技术 (micro packaging technology)

以微电子、高密度组装和微焊接等技术为基础,在多层布线基板上,将微电子器件及微型元件组装成电子硬件的一种工艺技术。它涉及固态技术、薄膜技术、厚膜技术、微电路技术、互连与微焊接技术、热控制技术、高密度组装技术、测试技术、可靠性技术和计算机辅助工程等领域,是一门电路、结构、工艺、材料、元器件等紧密结合的综合性技术。

电子组装技术经历了以下几个阶段: ①20 世纪 40 年代是以电子管为有源器件的手工焊接阶段。②40 年代晶体管和印制电路相继问世,并在 50 年代至 60 年代得到广泛应用后,形成了以晶体管和印制电路板为主的手工焊接阶段。这阶段电子设备的组装和结构产生了很大的变化,提高了组装密度,缩小了设备的体积。③60 年代,在集成电路技术与多层印制板的发展的基础上,形成了以集成电路、自动插装和波峰焊为主的组装阶段。④70 年代末,由于超大规模集成电路和无引线或短引线片状电子元器件的发展,电子组装进入了表面安装阶段。⑤80 年代中期,在发展表面安装技术的同时,微焊接技术、高密度多层基板技术的发展,形成了以多芯片模块 (MCM) 为特征的第五代微电子组装阶段。其特点是组装密度更高,互连线更短,因此,信号延迟时间短,信息传输速度快。随着组装工艺及材料科学的不断发展,微组装又向三维立体组装发展。目前三维组装主要有两种形式:一种是在氧化铝基板上先

形成薄膜电路,然后敷上一层聚酰亚胺薄膜,再在此薄膜上镀镍或铜,以形成带状薄膜电缆所需的布线图形,而后用再流焊将倒装片集成电路、片式电容器和片式电阻器等焊在此薄膜电缆上。另一种形式是将电容和电阻等埋入多层陶瓷基板的内部,利用同时烧成技术,使薄膜电缆、电子元件埋藏在基板的层间。三维组装芯片间的互连线更短,因此,减小了芯片间的阻容负载和信号延迟时间,也减小了寄生效应。

微组装与常规电子组装的主要区别在于所用的组装结构和互连技术不同。常规的电子组装是以一般电子元器件及普通印制电路板为基础的组装技术,微组装则是以集成电路和高密度多层基板以及微焊接为基础的综合性组装技术。微组装能减小电子元件和芯片的安装面积及互连线的长度,并能扩大基板尺寸和布线层数,以容纳更多的集成电路芯片和元器件,从而提高其组装密度,完成更多更复杂的电路功能。

高密度多层布线技术是微组装中缩小组装器件(或设备)的体积,减小布线总长度,缩短信号延迟时间,提高信息传输速度的一项关键技术。一般采用薄膜和厚膜技术布线,要求布线结构微细,布线长度尽可能地短,还要求线间的介电常数小。通常有下列 4 种布线技术。

(1) 多层陶瓷布线 在低温烧结玻璃陶瓷多层布线基板上,用光刻技术形成金或银-钯薄膜布线。在底层基板上,用生带技术形成电源层和接地层布线,而上层基板上则是信号层布线,在各层陶瓷表面上溅射钽、钯两层膜,经光刻、显影形成布线图形。

(2) 多层薄膜聚酰亚胺布线 在陶瓷或硅基板上电镀或溅射 Cr/Cu/Cr 电源层,敷上聚酰亚胺薄膜,在其上再涂光致抗蚀剂,并制作通孔图形,以抗蚀剂为保护膜,腐蚀聚酰亚胺,形成互连通孔。除去抗蚀剂后,在表面溅射几微米厚的 Cr/Cu/Cr 层,用光刻法形成第一层信号布线层,再涂敷介质聚酰亚胺层,形成互连通孔,然后按上述方法形成第 2 信号布线层和接层等等。这种布线技术已达 63 层。

(3) 厚膜布线 用微型笔或浆料喷射法,直接在厚膜导体层上按设计要求形成微细布线图形,布线材料一般用铜。

(4) 混合布线 是一种厚、薄膜结合的布线技术,用标准厚膜工艺形成电源层和接地层布线,用光刻光形成互连通孔,再在上面用薄膜工艺形成信号层布线。这种布线技术的组装密度高于上述几种布

线技术,陶瓷基板尺寸可减小几倍,适用于高频多芯片模块的组装,能降低信号衰减和串扰,成本也比较低。目前国际上正在研究利用超导技术进行布线,超导布线可将 40~60 层布线减小到只有 4 层布线(电源层、接地层、 x 层和 y 层)结构。

改进芯片及元器件的互连和安装方法是缩小体积、提高组装密度和可靠性的一项重要途径。目前在微组装中常用的微焊接方法有丝焊、激光微焊、芯片基板焊、倒装焊和载带自动焊等。丝焊和倒装焊属直接安装法,安装面积小,但芯片不能老化筛选和预测,影响混合电路或微电子组装组件的合格率和可靠性。芯片基板焊是一种将集成电路或芯片直接安装与互连到印制电路板上的焊接技术。载带自动焊是焊接集成电路内、外引线的一种自动群焊法,它先把载带(镀锡或镀金的铜箔、粘接剂塑料膜制成的具有引线框的柔性印制电路板)用热脉冲焊或超声热压焊焊到芯片焊区的镀金凸台上(称内引线焊接),冲剪下此载带,并迅速将其焊到基板的焊盘上(称外引线焊接),整个焊接过程自动完成。

由于微组装的组装密度很高,其体积功率密度(或面积功率密度)也很高,采取适当的热控制技术亦是微组装的一项重要内容。热控制的目的是为了尽量减小电子组件或设备的内部热阻和外部热阻,利用热传导、对流换热和辐射换热把热源的热量迅速散发至周围环境。常用的冷却方法有:自然冷却(包括传导冷却、自然对流冷却和辐射换热)、强迫空气冷却、液体冷却、蒸发冷却、热电致冷(半导体致冷)、热管传热等。冷却方法的选择主要取决于元器件或设备的发热功率密度及其允许的温升。采用散热性能好的基板材料和介质材料,改进元器件内部各电极的焊接质量,均可减小其内部热阻。

参考文献

Sinnadurai F N. Handbook of Microelectronic Packaging and Interconnection Technologies. Scotland: Electrochemical Publications, 1985 (赵悼爻)

weihu gongju

维护工具(maintenance tool) 用于软件维护过程中的软件工具。它辅助维护人员对软件代码及其文档进行各项维护活动。维护工具范围很广,如版本控制工具、文档分析工具、开发信息库工具、逆向工程工具和再次工程工具等。

UNIX 的 SCCS(源代码控制系统)是版本控制工具的典型代表。SCCS 能为正文文件建立一个版

本树,第一版完整储存文件全文,以后各版只存放它同前续版本的不同之处,在任意时刻 SCCS 只允许对一个当前版本进行修改、提交。通过版本树维护了版本更新历史,并允许恢复以前的版本。

文档分析工具用于对软件开发过程中的各种形式的文档进行分析,给出进行软件维护活动所需要的维护信息,如基于 DFD 图的需求文档分析工具可给出对某个加工进行维护时的影响范围及被影响范围分析;基于模块结构图的设计文档分析工具可以给出对模块、变量进行维护的影响及被影响范围分析。

开发信息库工具用来维护软件项目的开发信息,包括对象、模块和子程序等。它记录每个对象的修改信息(已确定的错误及重要改动)和其他变形(如抽象数据结构的多种实现),还必须维护对象和与之有关信息之间的关系,包括模块的设计者,新一版中模块的改动及其与错误、测试实例、测试结果之间的联系等。其他必须记录的信息包括:用来生成此软件产品的所有工具的版本信息(如编译程序、连接程序、程序生成程序的版本号),所采用的命令语言程序和系统库以及测试实例版本和测试报告。

逆向工程工具用以通过对源程序的分析,恢复设计信息,如程序的体系结构、控制结构、逻辑结构、数据结构和数据流等。逆向工程工具可分为两类:一类是动态的,即通过监控一个程序的执行来建立此程序的行为模型,此类工具较少;另一类是静态的,即从分析源程序出发来抽取各种程序结构,包括控制流、逻辑流、数据结构和数据流,以便恢复设计信息。

再次工程工具用来支持重构一个功能和性能更为完善的软件系统。目前的再次工程工具主要集中在代码重构、程序结构重构和数据结构重构等方面。代码重构工具把用一种程序语言书写的软件转换成用另一种语言书写的或适用于不同硬件平台的软件。程序结构重构工具接受一个非结构化或结构化程度较低的源程序,构造出行为等价的结构化程序。数据结构重构工具通过对数据描述的分析,重构新的数据结构。

参考文献

Arnold R S. Software Reengineering. Los Alamitos: IEEE Computer Society Press, 1993

(钱乐秋 赵文耘)

weihu guocheng

维护过程 (maintenance process) 软件维护人员所负责的一系列活动,其目的是在保持软件整体性能的同时对它进行修改,使它达到某一需求,直到其退役才告终止。从维护方式上讲有三种维护:改正性维护、适应性维护以及改善性维护。当软件由于错误、缺陷、问题需要改进和修改,以及对相应文档进行修改时,都要涉及维护过程。

维护过程包含的活动有:问题分析和修改分析、修改和实施、对维护的评审和验收、移植、软件退役等。维护过程同时还贯穿软件过程中其他过程的实施:维护人员通过**管理过程**管理维护过程;通过**剪裁过程**剪裁维护过程中的活动;通过改进过程参与维护过程的管理(参见**软件过程**);通过**支持过程**实施维护过程中的文档编制、评审、质量保证等。当进行某个活动时,维护过程可能需进入开发过程(如在实施软件的修改时),这时维护人员即是那里的开发人员(参见**开发过程**)。

以下是维护过程所涉及的活动,其中活动(1)是必需的,且是应当首先完成的。活动(2)至活动(6)为可选的,开发人员应当首先完成活动(1)的任务,然后根据活动(1)所产生的结果选择活动(2)至(6),完成维护过程。

(1) 本过程的实施准备 目的是为维护过程准备最基本的约定。其主要任务有:①制定活动(2)至活动(6)的实施计划和步骤;②确定对用户的问题报告和修改请求进行接收、跟踪的步骤以及向用户反馈信息的方式和步骤;③指定文档编制方式、配置管理方式以及各支持过程的实施方法。

(2) 问题分析和修改分析 目的是分析软件修改将对系统、接口带来的影响并确立修改方案。主要任务有:①分析维护类型,是改善型、改正型,还是适应新环境型的维护;分析维护的范围,包括修改规模、成本、时间;分析维护对关键问题(如性能、保密性)的影响;②在分析的基础上,选择实施修改的方案。

(3) 修改的实施 目的是对问题及修改进行更为详细的分析,并实施修改。主要任务有:①详细分析问题报告和修改请求,决定哪些文档、软件单元和版本需要修改;②实施修改。此时维护人员需进入**开发过程**完成修改。开发过程中的需求应作出相应的修改,最低要求是保证未经修改的需求不受影响,而新的修改过的需求得到完全、正确的实现。

(4) 对维护进行评审和验收 此活动任务是:

维护人员同授权修改的机构一起进行评审,评定经过修改之后的系统的整体性能。

(5) 移植 目的是将一个系统或软件从一个旧的运作环境移植到一个新的运作环境中,并保证该移植活动的正确性。由于涉及软件的修改,因而也是维护活动。其主要任务有:①制定移植计划并执行该计划。计划中至少包括:对需求分析和移植的定义;移植工具的开发问题;软件和数据转换问题;移植计划的执行问题;移植的验证问题;以后对旧环境的支持问题等等;②向用户通告移植计划和移植执行情况;③旧环境和新环境最好并行运行,并向用户提供必要的培训;④对移植活动及移植后的结果进行评审。

(6) 软件退役 软件将根据所有者的要求退役。此时,维护人员应当:①制定软件支持的撤销计划,并执行该计划。计划中至少包括:在多长时间后全部或部分地停止软件支持;系统及有关文档如何存档;若以后仍需要支持时的责任问题;若需要,转移到新的软件上的问题。②向用户通告退役计划及执行情况。③将退役软件的所有文档、记录、数据归档。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes — IEEE Std. 1074 ~ 1991
2. ISO /IEC 12207:1995 Information Technology — Software — Part 1: Software Life -Cycle Process

(宿为民)

weiyena kaifa fangfa

维也纳开发方法 (Vienna development method, VDM) 20 世纪 70 年代由 IBM 维也纳实验室的 C. Jones 和 D. Bjorner 提出的,最初是作为描述程序设计语言**指称语义**的元语言,后来发展成支持程序开发的形式化方法。

VDM 基于集合论,其出发点是用一阶谓词(断言)来描述程序的状态空间(程序状态即程序中所使用的全体非局部变量的取值)。VDM 提供了四个基本类型:集合、复合、映射和序列,作为书写规约的基础。每个基本类型都具有相应的运算和关系,可以在断言中使用。程序的功能由刻画其初始状态的前断言及刻画其终止状态的后断言规定。由于程序的功能描述往往涉及到某些变量在其执行后与执行前的取值之间的关系,在后断言中可以用特定的符号来引用变量在程序执行前的初值。VDM 采用“面

向模型”的方法来描述数据类型(参见形式规约)。一数据类型的规约由三要素组成:①状态集;②初始值;③各运算的描述。每个运算均用前、后断言描述。

例 先进先出队列的 VDM 规约

Queue = seq of Qel

$q_0 = []$

ENQUEUE(e ; Qel)

ext wr q: Queue

post $q = \tilde{q} \frown [e]$

DEQUEUE () e: Qel

ext wr q: Queue

pre $q \neq []$

post $\tilde{q} = [e] \frown q$

这个例子利用序列来给出队列的规约。其中 $[]$ 表示空序列。 $[e]$ 是仅含 e 的单元元素序列, \frown 连接两个序列。保留字 pre, post 分别引出前、后断言。出现在后断言中的 \tilde{q} 表示变量 q 在程序开始执行时的值(即前断言中的值)。保留字 ext wr 表示随后的变量是外部的,其值在本程序中被引用(r)且被修改(w)。施加于 Queue 上的运算有两个:ENQUEUE 和 DEQUEUE。ENQUEUE 不带前断言,表示其前断言为真。

VDM 提供了一套变换规则。按这些规则可以将规约精化,逐步转换为可执行程序,并保证最终得到的程序满足起初的规约。

VDM 的提倡者十分重视将其应用于工业规模的软件开发实践。他们与工业界联合举办 VDM 研讨班,训练系统设计员和程序员掌握、使用这种方法。设计国际标准化组织(ISO)规约语言 VDM-SL 的工作目前正在进行之中。

参考文献

Jones C B. Systematic Software Development Using VDM. New York: Prentice - Hall, 1986

(林惠民)

wei suijishu

伪随机数 (pseudo-random numbers) 在数字计算机上用数学方法产生的,具有 $[0,1]$ 区间上均匀分布母体性质的数值序列 $\{r_n; n = 1, 2, \dots, 0 \leq r_n \leq 1\}$ 。在一台 b 进制,尾数字长为 k 位的计算机上,任取 m 个整数, x_1, \dots, x_m 作为初值,按某种递推

公式 $X_{n+m} = G(x_n, \dots, x_{n+m-1})$,便可产生一个数列,如果经过统计假设检验,表明这个数列具有 $[0,1]$ 均匀分布母体的样本性质,就称其为 $[0,1]$ 均匀分布的伪随机数列。这里“伪”的意义是,它本身并不是真正的随机数,而且总有周期性。我们希望在满足统计性能的前提下,周期尽可能地长。平方取中法是产生伪随机数列最早使用的方法,它将一个 $2s$ 位十进制随机数,平方之后截取中间的 $2s$ 位作为新的随机数,重复上述过程便得到一个伪随机数列。但按此法所产生的伪随机数列有退化的危险,即出现的数字都变为0或者形成重复周期变化的序列,且周期难以确定。此外有取中法和移位法,但结果不理想。目前产生伪随机数列比较好的方法是同余法,包括加同余法、乘同余法和混合同余法。

加同余法的公式为: $x_{n+2} = x_n + x_{n+1} \pmod{M}$,
 $r_{n+2} = x_{n+2} / M$ 。但该数列的相关系数太大(约为 -0.3),不宜直接作为伪随机数列。

混合同余法的公式为:

$$x_{n+1} = \lambda x_n + c \pmod{M},$$

$$r_{n+1} = x_{n+1} / M \quad n = 0, 1, 2, \dots$$

其中 x_0 为初值, λ 为乘子, c 为增量, M 为模。 $c = 0$,即为乘同余法。一般取 $M = b^k$, x_0 与 c 的选取对数列的性质有很大的影响。用数论的方法可证明:混合同余数列达到周期 M 的充要条件是:① c 与 M 互素;②对每一个 M 的素因子 p , $\lambda - 1$ 为 p 的倍数;③若 M 是4的倍数,则 $\lambda - 1$ 是4的倍数。若 $b = 2, M = 2^k$,应取 $\lambda = 4q_1 + 1, c = 2a_1 + 1, x_0$ 为非负整数,其中 q_1, a_1 为正整数。由上述可见,乘同余法不能达到最大周期 M ,但可以证明乘同余法最大可能周期为 $2^{k-2} (k > 2)$ 。如取 $\lambda = 8a + 3, x_0 = 2b + 1 (a, b \text{ 为任意正整数})$,且 λ 的二进制表示中0,1的出现不规则时,便可得到周期为 2^{k-2} 且统计性质较好的伪随机数 $\{r_n = x_n / 2^k, n = 1, 2, \dots\}$ 。

通过 $[0,1]$ 均匀分布的伪随机数 $\{r_n\}$,便可得到各种不同分布的伪随机数。事实上有定理:设随机变量 η 的分布函数 $F(x)$ 连续,且反函数 $F^{-1}(x)$ 存在,则 $R = F(\eta)$ 便是 $[0,1]$ 上均匀分布的随机变量。因此, $\{\eta_n = F^{-1}(r_n); n = 1, 2, \dots\}$ 为 η 的抽样序列。例如 η 为指数分布, $F(x) = 1 - e^{-\lambda x} (x > 0)$,则 $\{\eta_n = -\frac{1}{\lambda} \ln(1 - r_n)\}$ 便是 η 的抽样序列。

如果 η 为 $N(0,1)$ 分布,由于 $F(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}}$

$\times e^{-(x^2/2)} dx$ 的反函数没有显式表示, 所以不能用上述方法求 η 的抽样序列。通常采用两种方法: ①由中心极限定理, $\eta_n = \left(\sum_{i=1}^n r_i - \frac{n}{2} \right) / \sqrt{\frac{n}{12}}$ 有渐近正态 $N(0, 1)$ 分布。一般取 $n = 12$, 则可用 $\{\eta_n: n = 1, 2, \dots\}$ 作为 $N(0, 1)$ 的抽样序列; ②坐标变换法: 设 r_1, r_2 为两个相互独立的 $[0, 1]$ 均匀分布的随机变数, 作变换

$$\eta_1 = (-2 \ln r_1)^{1/2} \cos(2\pi r_2)$$

$$\eta_2 = (-2 \ln r_1)^{1/2} \sin(2\pi r_2)$$

则可证明 η_1, η_2 是两个独立的 $N(0, 1)$ 分布的随机变数。

产生其他分布随机序列的方法还有舍选法、离散近似法、复合抽样等。但不管用什么方法产生的伪随机数列, 都必须进行统计检验以确认它们有良好的统计性质。这些检验有参数检验、分布均匀性检验、独立性检验等。

参考文献

1. 中山大学数力系. 概率论与数理统计(下册). 北京: 人民教育出版社, 1980
2. 徐钟济. 蒙特卡罗方法. 上海: 上海科学技术出版社, 1985 (金治明)

weixing jieru jishu

卫星接入技术 (satellite technologies) 利用卫星通信提供宽带接入的技术。对卫星技术而言, 最令人感兴趣的是极高频 SHF 波段。国际电信联合会 (ITU) 为卫星传输分配的波段是 2.5 GHz 到 22 GHz。这是十分高的频率, 相应的波长极短, 因此称它为微波。微波在卫星和地面站之间沿直线直接

传输。其优点是这么高的频率不会被地面电离层所阻挡, 这意味着信号发送到卫星, 再返回到地面站接收不会引起很大衰减。

在 20 世纪 60 年代, 第一个卫星运行在 C 波段, 由于这个频率范围覆盖了很多地面微波链路, 因此必须限制 C 波段卫星的功率以免干扰地面的微波链路。之后在 70 年代, 开始出现 Ku 波段的卫星, 由于 Ku 波段的频率范围不覆盖地面微波, 因此可运行比 C 波段更高强度的信号。例如, C 波段卫星通常发送在 33 ~ 38 dBW (瓦特分贝, 相对于 1 瓦的分贝), 而 Ku 波段卫星则发送 47 ~ 52 dBW。Ku 波段的另一个优点是因为运行在更高的频率, 波长更短, 因此接收无线的波束更窄。高发送功率和窄波束使卫星成本更低, 效率更高, 因而便于推广。

参考文献

1. 胡道元. 智能建筑计算机网络工程. 北京: 清华大学出版社, 2002
2. Padmanand Warriar, Balaji Kumar, XDSL Architecture. McGraw-Hill, Inc., 2000 (胡道元)

weixing tongxin

卫星通信 (satellite communication) 利用人造地球卫星作为中继站转发无线电波, 在两个或多个地球站或地面站之间进行的通信。是无线电通信的形式之一。卫星通信是在地面微波中继通信和空间技术的基础上发展起来的。因为微波是直线传播的, 微波接力通信是一种“视距”通信, 即只在“看得见”的范围内才能通信。由于地球是圆形的, 通信卫星的作用相当于离地面很高的中继站, 如图 1 所示。

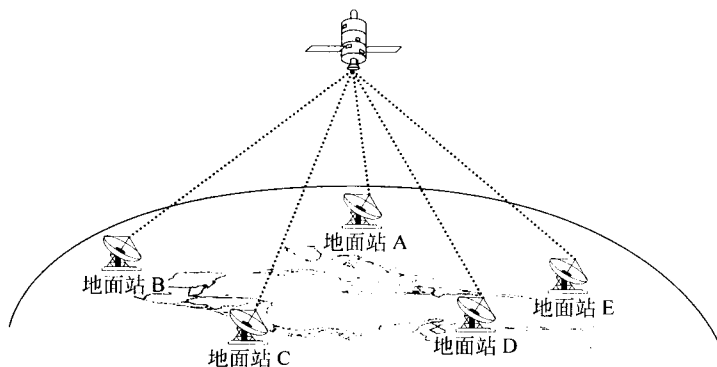


图 1 卫星通信示意图

卫星通信系统按照其使用的空间轨道位置,可分为对地静止轨道(GEO)系统和非对地静止轨道(Non-GEO,包括MEO、LEO)系统;按照其业务提供的范围可分为全球卫星通信系统和区域卫星通信系统。低轨道卫星(LEO)高度一般为500~1500 km左右,中轨道卫星(MEO)高度通常为5000~15000 km左右,GEO为35768 km高度赤道上空的轨道。

卫星通信与其他通信方式相比,具有下列特点:①通信覆盖区域大,通信距离远;②以广播方式工作,便于实现多址连接;③通信容量大,能传送的业务类型多;④通信机动灵活,不受地理位置的限制;⑤通信质量好,可靠性高;⑥通信的成本与距离无关。

卫星通信在技术上也遇到了一些新的问题,主要是:①需要先进的空间和电子技术;②需要解决信号传播时延带来的影响;③需要圆满实现多址连接,解决多址技术问题;④需要保证卫星高稳定地和可靠地工作。

以对地静止轨道卫星通信系统为例。一个卫星通信系统由空间分系统、通信地球站、跟踪遥测及指令分系统和监控管理分系统等四大部分组成。跟踪遥测及指令分系统的功能是对卫星进行跟踪测量,控制其准确进入静止轨道上的指定位置,待卫星正常运行后,要定期对卫星进行轨道修正和位置保持;监控管理分系统的功能是对定点的卫星在业务开通前、后进行通信性能的监测和控制;空间分系统即通信卫星,由温控、能源、控制、通信、天线、跟踪、遥测及遥控等组成,其中天线和通信(即转发器)称为卫星的有效载荷,其余组成部分合起来称为卫星公共舱,为维持有效载荷在空中正常工作起保障作用;通信地球站是卫星通信系统的重要组成部分,由天线、馈线设备、发射设备、接收设备、信道终端设备、电源设备以及监控设备组成。

卫星通信网是基于卫星通信的一种广域网,网络结构主要有两种:星状和网格形。

一个典型的卫星通信线路的构成如图2所示。

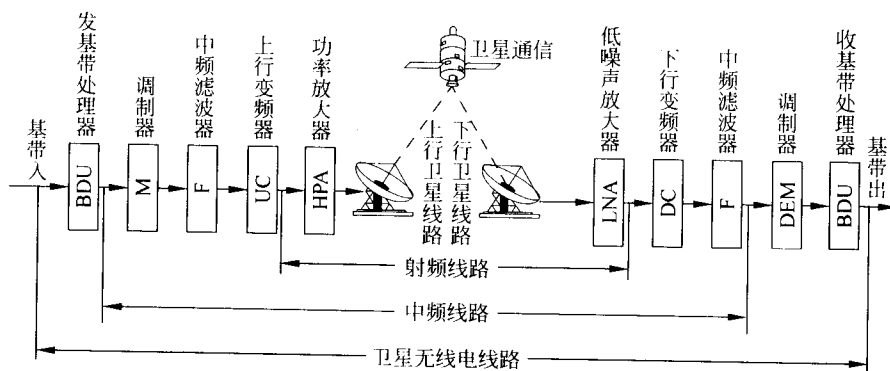


图2 卫星通信线路的构成

在卫星通信中,工作频段的选择是一个十分重要的问题。它直接影响到整个卫星通信系统的通信容量、质量、可靠性、设备的复杂程度和成本的高低,并还将影响到与其他通信系统的协调。卫星通信工作频段的选择,主要考虑下列因素:①电波应能穿过电离层,传播损耗和外部附加噪声应尽可能的小;②应具有较宽的可用频带;③合理使用无线频率资源,防止各种空间和地面通信业务间的相互干扰;④电子技术与器件的发展状况,现有通信技术设备的利用和配合。国际电信联盟(ITU)主持召开的1963年无线电特别行政会议(EARC)和1971年有关空间通信的世界无线电行政会议(WARC)规定了空间使用的频率分配原则。卫星通信系统常用的工

作频率主要有:①甚高频波段 UHF200~400 MHz;②L波段 1.5~1.6 GHz,主要用于移动卫星通信、海事卫星业务等;③C波段 4.0~6.0 GHz,主要用于固定卫星业务和专用卫星业务、甚小口径天线地球站(VSAL)网络等;④X波段 7.0~8.0 GHz,主要用于固定卫星业务;⑤Ku波段 11.0~14.0 GHz,主要用于VSAT网络、卫星电视广播、移动卫星通信等;⑥Ka波段 20.0~30.0 GHz,主要用于VSAT网络、卫星电视广播。

卫星通信的体制主要有频分多址(FDMA)、时分多址(TDMA)、卫星交换的时分多址(SS-TDMA)、码分多址(CDMA)、单载波多路(MCPC)、上行频分多址-下行时分多路(FDMA/TDM)、上行码分多址-

下行时分多路(CDMA/TDM)等。

参考文献

1. 吕海寰,蔡剑铭,甘仲民等. 卫星通信系统. 北京:人民邮电出版社,1988

2. 刘国梁,荣昆璧. 卫星通信. 西安:西安电子科技大学出版社,1990 (史美林 英春)

wendang yuyan

文档语言 (documentation language) 用于书写计算机软件文档的语言。计算机软件文档是计算机软件开发、维护和使用过程的档案资料和对软件本身的阐明性资料。

根据文档的作用和性质,常见的计算机软件文档可分为:

- (1) 对软件本身的描述,如软件**基准手册**;
- (2) 对软件使用方式的描述,如用户指南,程序人员指南,联机帮助等等;
- (3) 软件开发过程的记录和中间产品与结果,如需求定义,功能规约,设计规约,测试计划,测试报告,维护与修改记录等等。

书写前两种软件文档的语言主要是自然语言,辅以图表等等。这些文档要求语言简练准确,通俗易懂,还经常以一定的结构储存在计算机系统中,如以超文本的形式,以便用户联机检索。

对于书写软件开发过程中产生的中间结果的语言,则通常根据中间结果的特性使用不同的语言。如,需求定义常采用E-R图,状态转移图,数据流图等,加上自然语言描述(参见需求定义语言)。书写软件功能规约的语言既可是形式化的功能规约语言,如Z语言,OBJ语言等等,也可是自然语言。软件的设计规约可使用特定的软件设计语言书写(参见设计规约),也可采用软件结构图加上自然语言来描述。良好的测试计划,维护与修改记录等应当是按照标准的格式使用自然语言书写。软件测试报告通常以自然语言填写的表格的形式出现。

(朱鸿 金凌紫)

wenfa

文法 (grammar) 语言结构的一种有限描述。给定任意的有限字母表 Σ , Σ^* 表示由 Σ 中的字母组成的所有符号串(包括空串)的集合。 Σ^* 的每个子集都是 Σ 上的一个语言。形式语言主要研究这种界限明确的语言。N. Chomsky 于1959年提出的生

成文法是形式语言的一种描述手段。形式上,生成文法是一个四元组

$$G = (\Sigma, V, S, P)$$

其中 V 是有限的变量集合,又称为非终结符号表; Σ 是有限的字母集合,又称为终结符号表, $V \cap \Sigma = \emptyset$; S 是开始符号, $S \in V$; P 是有限的形如 $\alpha \rightarrow \beta$ 的生成式集合, $\alpha \in (V \cup \Sigma)^* V (V \cup \Sigma)^*$, $\beta \in (V \cup \Sigma)^*$ 。生成式又称为规则。在不对文法 G 的生成式增加限制时,称文法 G 为无限制文法,或者短语结构文法,或者0型文法。

推导 由文法生成语言的句型的过程。给定文法 $G = (\Sigma, V, S, P)$ 和 $\alpha', \beta' \in (V \cup \Sigma)^*$,如果 $\alpha' = \beta'$ 或者存在 $\alpha_1, \alpha_2, \alpha, \beta \in (V \cup \Sigma)^*$,使得 $\alpha' = \alpha_1 \alpha \alpha_2$, $\beta' = \alpha_1 \beta \alpha_2$ 且 $(\alpha \rightarrow \beta) \in P$,则称在文法 G 中 α' 直接推导为 β' ,记为 $\alpha' \Rightarrow \beta'$ 。用 $\stackrel{\cdot}{\Rightarrow}$ 表示 \Rightarrow 的自反传递闭包。如果 $S \stackrel{\cdot}{\Rightarrow} \alpha, \alpha \in (V \cup \Sigma)^*$,则称 α 为文法 G 生成的句型。文法 G 生成的所有的句型的集合

$$S(G) = \{\alpha \in (V \cup \Sigma)^* \mid S \stackrel{\cdot}{\Rightarrow} \alpha\}$$

例如短语结构文法 $G_1 = (\{a, b\}, \{S, A, B, C\}, S, P_1)$,其中 $P_1 = \{S \rightarrow ASa, S \rightarrow BSb, S \rightarrow C, AC \rightarrow aC, BC \rightarrow bC, Aa \rightarrow aA, Ba \rightarrow aB, Ab \rightarrow bA, Bb \rightarrow bB, C \rightarrow \lambda\}$,则有推导

$$S \xRightarrow{c_1} ASa \xRightarrow{c_1} ABSba \xRightarrow{c_1} ABCba \xRightarrow{c_1} AbCba \xRightarrow{c_1} bACba \xRightarrow{c_1} baCba \xRightarrow{c_1} baba$$

这里 $S, ASa, ABSba, ABCba, AbCba, bACba, baCba$ 和 $baba$ 都是 G_1 生成的句型。

短语结构语言 短语结构文法生成的所有句子的集合。如果文法 $G = (\Sigma, V, S, P), \alpha \in S(G)$ 且 $\alpha \in \Sigma^*$,则 α 是文法 G 生成的句子。文法 G 生成的语言

$$L(G) = S(G) \cap \Sigma^* = \{w \in \Sigma^* \mid S \stackrel{\cdot}{\Rightarrow} w\}$$

0型文法生成的语言称为0型语言或者短语结构语言。例如在上面给出的文法 G_1 中, $S \stackrel{\cdot}{\Rightarrow} baba$,故 $baba \in \Sigma^*$ 是 $L(G_1)$ 的一个句子。实际上,文法 G_1 生成的短语结构语言 $L(G_1) = \{uu \mid u \in \{a, b\}^*\}$ 。

参考文献

1. 霍普克洛夫特 J E,厄尔曼 J D 著. 自动机理论、语言和计算导引. 徐美瑞译. 北京: 科学出版社, 1986

2. Chomsky N. On Certain Formal Properties of Grammars. Information and Control, 1959, 2(2): 137 ~ 167 (郭清泉)

wenhua chengxu sheji

文化程序设计 (literate programming) 一种程序设计方法,将程序设计视为著述活动,目标是使程序员写非常容易理解的软件。它提供软件支持,允许程序员完全按照自己思路的本来逻辑顺序来写程序,并且同时产生可执行代码及具有高可读性的程序说明文件。

这种设计方法主张,应该把程序看成具有网状结构:一个较复杂的软件片段是由若干个较简单的片段和它们之间的简单关系所构成的,每一个这样的片段自然又可以由更简单的片段来构成。程序员的任务是以一种最有利于人们理解的次序来表述这些片段和其关系,而不是固定地按照某种确定次序,例如自顶向下或者自底向上。这样,无论是软件作者还是读者,都得到可理解的软件。

文化程序设计首先由 D. E. Knuth 于 20 世纪 80 年代所倡导,并在 1983 年研制出第一个文化程序设计系统 WEB。WEB 语言所提供的设施允许人以“意识流”的次序来表达程序,程序员可以把大程序看成一张网,以一种从心理上看来是正确的次序来探索它。WEB 的语言包含了两种成分,一种是供描述算法过程的 **PASCAL 语言**,另一种是描述程序文档排版要求的 **TEX 排版语言**。

WEB 系统由两个子系统组成,一个子系统从 WEB 语言程序中自动地抽出描述算法过程的部分,并且加工成 PASCAL 编译程序所能接受的形式,然后据此得到可以在计算机上执行的代码。另一个子系统则是把 WEB 语言程序加工为 TEX 系统所能接受的形式,并据此得到具有高度可读性的程序说明文件。文化程序设计的两种组成语言是可以更换的。例如曾经有 **C 语言**和 **TROFF** 的组合,C 和 **TEX** 的组合 (**CWEB**),C 和中西文排版语言 **SP** 的组合 (**CDS**)。被采用过的其他程序语言还包括 **Modula-2**,**FORTRAN**,**Ada** 等。

有几个软件系统,即 **WEB** 和 **CWEB** 本身,**TEX** 和 **METAFONT**,已经被 D. E. Knuth 等人用 **WEB** (或者 **CWEB**) 写出,并且公开出版。它们也都被列为自由软件,任何需要的人均可以从因特网上获取。

下文的图 1 是 quick sort 算法的一个完整程序文档(为紧凑起见,消除了空白,页面用水平线隔离)。它是图 2 中的 **CWEB** 源程序经系统加工而得到的。

QSORT

	Section	Page
Quick sort function for array	1	1
Index	10	3

§ 1. QSORT

QUICK SORT FUNCTION FOR ARRAY 1

1. **Quick sort function for array.** This function sorts or partially sorts an array into increasing order. Every element of the array has a key which is comparable, and sorting is according to the value of the keys. After this sorting process, the elements are rearranged and the keys of the elements are in increasing order.
2. This function uses "Quick sort" algorithm. This algorithm uses the divide and conquer technique. To begin each iteration an element is selected from the file and the file is then split into two subfiles, those elements whose keys are smaller than the selected one and those elements whose keys are larger. In this way, the selected element is placed in its proper final location between the two resulting subfiles. This production is repeated recursively on the two subfiles and so on. Quick sort is a very popular sorting algorithm; although its worst case is $O(n * n)$, its average performance is excellent. This worst case occurs when the file is in order already. Any portion of the file that is nearly in order will significantly deteriorate Quick sort's efficiency.
3. Action definitions are related to what the element type is defined. **KEY**(*elemt*) is the key of the element. **Assign**(*elemt1*,*elemt2*) sets the value of *elemt1* to be the one of *elemt2*.


```
#define KEY(elemt) elemt
#define Assign(elemt1,elemt2) elemt1 ← elemt2;
```
4. The program structure is here


```
< Type definitions 5 >
< Function 6 >
```

5. **ArrayEntry** is the type name of the elements of the array to be sorted.

ArrayToSort is the type name of the array to be sorted.

< Type definitions 5 > ≡

typedef int ArrayEntry;

typedef ArrayEntry ArrayToSort[];

This code is used in section 4.

6. Now is the Quick sort function. *r* is the array to be sorted, *lo* and *up* are the lower and upper bound, respectively, of array *r*.

< Function 6 > ≡

**sort (ArrayToSort r, int
lo, int up)**

{

< Local variables 7 >

< Iteration loop 8 >

}

This code is used in section 4.

7. We need a temporary unit *temp* to hold the selected element, also two variables *i* and *j* to point to currently processed elements.

< Local variables 7 > =

int i, j;

ArrayEntry temp;

This code is used in section 6.

§ 2. QUICK SORT FUNCTION FOR ARRAY

QSORT § 8

8.

< Iteration loop 8 > =

while (up > lo)

{

i ← *lo*;

j ← *up*;

Assign(temp, r[lo]);

(Split file in two, elements whose keys are smaller than *temp* are in *r*[*lo* . . *i* - 1], and ones with larger keys are in *r*[*i* + 1 . . *up*])

Assign(r[i], temp); / * *i* is the correct location for temp * /

sort(r, lo, i - 1); / * Sort recursively * /

lo ← *i* + 1;

}

This code is used in section 6.

9.

< Split file in two, elements whose key are smaller than *temp* are in *r*[*lo* . . *i* - 1], and ones with larger keys are in *r*[*i* + 1 . . *up*] > ≡

While(i < j)

{

While(KEY(r[j]) > KEY(temp))

j ← *j* - 1;

r[*i*] ↔ *r*[*j*];

While(i < j ∧ KEY(r[i]) ≤ KEY(temp))

i ← *i* + 1;

r[*j*] ↔ *r*[*i*];

}

This code is used in section 8.

§ 10 QSORT

INDEX

3

10. Index.

ArrayEntry: 5, 7.

ArrayToSort: 5, 6.

Assign: 3, 8.

elemt: 3.

elemt1: 3.

elemt2: 3.

i: 7.

j: 7.

KEY: 3, 9.

lo: 6, 8.

r: 6.

sort: 6, 8.

temp: 7, 8, 9.

up: 6, 8.

4 NAMES OF THE MODULES

QSORT § 10

< Function 6 > Used in section 4.

< Iteration loop 8 > Used in section 6.

< Local variables 7 > Used in section 6.

< Split file in two, elements whose keys are smaller than *temp* are in *r*[*lo* . . *i* - 1], and with larger keys are in *r*[*i* + 1 . . *up*] > Used in section 8.

< Type definitions 5 > Used in section 4

图 1 Quick sort 算法的程序文档

@ * Quick sort function for array.

This function sorts or partially sorts an array into increasing order. Every element of the array has a key which is comparable, and sorting is according to the value of the keys. After this sorting process, the elements are rearranged and the key of the elements are in increasing order.

@ This function uses "Quick sort" algorithm. This algorithm uses the divide and conquer technique. To begin each iteration an element is selected from the file and the file is then split into two subfiles, those elements whose keys are smaller than the selected one and those elements whose keys are larger. In this way, the selected element is placed in its proper final location between the two resulting subfiles. This production is repeated recursively on the two subfiles and so on.

Quick sort is a very popular sorting algorithm; although its worst case is $O(n^2)$, its average performance is excellent. This worst case occurs when the file is in order already. Any portion of the file that is nearly in order will significantly deteriorate Quick sort's efficiency.

@ Action definitions are related to what the element type is defined.

|KEY (elemt)| is the key of the element.

|Assign (elemt1, elemt2)| sets the value of |elemt1| to be the one of |elemt2|.

@ C

#define KEY (elemt) elemt

define Assign (elemt1, elemt2) elemt1
= elemt2;

@ The program structure is here

@ C

@ <Type definitions @ >@ ;

@ <Function @ >@ ;

@ |ArrayEntry| is the type name of the elements of the array to be sorted.

|ArrayToSort| is the type name of the array to be sorted.

@ <Type... @ > =

typedef int ArrayEntry;

typedef ArrayEntry ArrayToSort[];

@ Now is the Quick sort function.

|r| is the array to be sorted, |lo| and |up| are the lower and upper bound, respectively, of array |r|.

@ <Fun... @ > =

sort(ArrayToSort r, int lo, int up)

{

@ <Local variables @ >@ ;

@ <Iteration loop @ >@ ;

}

@ We need a temporary unit |temp| to hold the selected element, also two variables |i| and |j| to point to currently processed elements.

@ <Local... @ > =

int i,j;

ArrayEntry temp;

@ @ <Iter... @ > =

while (up > lo)

{ i = lo;

j = up;

Assign (temp, r[lo]);

@ < split file in two, elements whose keys are smaller than |temp| are in |r[lo..i-1]|, and ones with larger keys are

in |r[i+1..up]| @ >

Assign (r[i], temp);

/* |i| is the correct location for |temp| */

sort (r, lo, i - 1); /*

sort recursively */

lo = i + 1;

```

    }
    @@ <split...@> =
    while (i < j)
    { while (KEY(r[j]) > KEY(temp))
        j = j - 1;
        r[1] = r[j];
        while (i < j && KEY(r[i]) < =
            KEY(temp))
            i = i + 1;
        r[j] = r[i];
    }
    @ * Index.

```

图2 Quick sort 算法的 CWEB 源程序

参考文献

1. Knuth D E. Literate Programming. The Computer Journal, 1984, 27(2): 97 ~ 111
2. Thimbleby H. Experiences of 'Literate Programming' using cweb (a variant of Knuth's WEB). The Computer Journal, 1986, 29(3): 201 ~ 211
3. Knuth D E, Levy S. The CWEB System of Structured Documentation. Version 3. 0. New York: Addison - Wesley, 1994 (董毓美 陈海明)

wenjian

文件 (file) 有组织的数据的集合。

早期,用户按物理地址存取存储媒体上的信息,使用不便、效率很低。引入文件概念后,用户不再需要了解文件存放的物理位置和物理结构,可实现“按名存取”,由文件管理程序根据用户给出的文件名自动地完成数据传输操作。把数据组织成文件加以管理是计算机数据管理的重大进展,其主要优点是:使用方便,安全可靠,便于共享。

从用户使用的角度看,文件可以分成两类(参见**文件管理程序**)。

在许多操作系统中,都把 I/O 设备看作是一个“文件”,称设备文件,这样用户无需考虑保存其文件的设备差异,用统一的观点去处理驻留在各种存储媒体上的信息,给使用带来极大方便。

可以各种方式对文件进行分类。按用途可分成:系统文件、用户文件和库文件;按保护级别可分成:只读文件、读写文件、可执行文件和不保护文件;按存放时限可分成:临时文件、永久文件和档案文

件;按设备类型可分为:磁盘文件、磁带文件、软磁盘文件;按信息流向可分成:输入文件、输出文件和输入输出文件。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989 (郑宇华)

wenjian chuansong

文件传送 (file transfer) 一台计算机在另一台计算机中按一定格式复制文件的技术。文件传送是计算机网络中最频繁的一种操作。互相传送文件的两台计算机可以是不同类型的,如微型计算机与大型计算机;也可以是安装不同操作系统的,如 DOS 与 UNIX。

要实现文件传送,必须在计算机上有相应的进程来启动,很多文件传送协议都允许文件的双向传送。例如,客户可以把文件发送给服务器,也可以向服务器请求文件并接收文件。为进行这样的传送,调用客户程序的用户必须给出自己的标识符,并且获得许可。本地许可由本地操作系统解决(如根据登录的标识符和口令在登录时获得许可)。访问远程文件的用户则必须得到对客户机授权的服务器的许可之后才能进行文件传送。

传送文件需要遵循一定的通信协议,以确保端对端传送的正确性。常用的文件传送协议有 XON / XOFF, Kermit, XMODEM, YMODEM, FTP, TFTP 和 FTAM 等,其中以 FTP 最为流行。

文件传送协议 FTP

文件传送服务是计算机网络中使用最广泛的应用之一。在因特网中,文件传送服务采用**文件传送协议 (FTP)**,因此,通常用 FTP 表示文件传送服务。通过 FTP,用户可与远程主机连接,直接将远程主机文件系统中的文件全部拷入本地文件系统,也可将本地文件全部拷进远地文件系统。在计算机网络中有成千上万个文件服务器供用户获取资源,包括公用程序、原始程序代码、研究报告、技术文档以及各类论文等。

FTP 是基于客户-服务器模型而设计的,与其他客户-服务器模型不同之处在于 FTP 客户与服务器之间要建立双重连接,一是控制连接,另一是数据连接。建立双重连接是因为 FTP 是交互式会话系统,客户每调用一次 FTP,便与服务器建立一次会话,该会话通过控制连接来维持,直至退出 FTP 为止。在一次 FTP 会话中,需建立一个控制连接和若干数据

连接。控制连接负责传送控制信息,尤其是客户命令(如文件传送命令等)。利用这种命令,客户可以向服务器提出多次请求。客户每提出一个请求,服务器即与客户建立一个数据连接,进行实际的数据(如文件)传送。一旦数据传送结束,数据连接相继撤销,但控制连接依然存在,客户可以继续发出命令,直到客户键入 close 命令才撤销控制连接,再输入 quit 命令,便退出 FTP 会话。

图 1 是 FTP 客户-服务器模型。为满足多个客户请求的需要,FTP 采用并发服务器方式。FTP 主服务器进程先于其他 4 个进程而存在,当客户发出 ftp 命令后,客户端首先建立客户控制进程,该进程建立自己的半控制连接,并向主服务器发出连接请求。主服务器接到客户连接请求后,创建服务器控制进程,该控制进程与客户控制进程建立控制连接,双方进入会话状态。这时主服务器进程退出,进入阻塞状态,等待新客户的 FTP 请求。

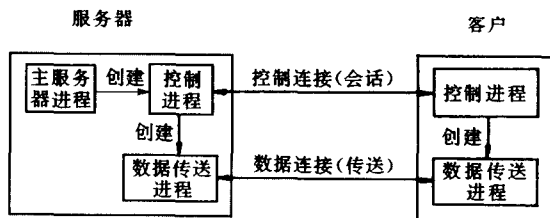


图 1 FTP 客户-服务器模型

控制连接建立后,客户控制进程向服务器发出数据请求,服务器控制进程收到请求后,创建服务器数据传送进程,该进程建立自己的半控制连接,并向客户控制进程发出连接请求。客户控制进程收到该请求后,创建客户数据传送进程,并与服务器传送进程建立数据连接,双方进行数据传送。传送结束后,撤销数据连接,数据传送进程终止。

FTP 的访问控制有两类:一类是严格的 FTP 访问控制,另一类是非严格的,前者要求客户给出文件所在的主机上的合法账号(包括注册名和口令),才能访问主机;后者由支持匿名 FTP 服务器提供的,便于广大客户获取主机中的公开文件,客户对这类服务器只要输入账号为 anonymous 和本地服务器所提示的口令即可。

通过计算机程序也可调用 FTP,这意味着文件传送能够自动进行。例如,可以设计一个应用程序在每天指定的时刻对某些文件进行检查,并且传送修改过的文件。运行这样的程序可以实现多种工作

的自动化。

文件传送、存取和管理协议 FTAM

FTAM 位于开放系统互连参考模型的应用层,它所提供的文件服务属于特定应用服务要素 SASE。

FTAM 标准定义了开放系统的文件服务,包括文件的传送、存取和管理。文件传送是一种使整个文件或部分文件内容在开放系统之间移动的功能;文件存取是对文件的部分内容进行检查、修改、替换或删除;文件管理是指生成或删除文件,以及检查或操作文件属性。

FTAM 文件服务采用了一种非对称的对话方式。通信双方按其功能分为与用户接口的发起方和与本地文件系统相联的响应方实体。文件服务由发起方主动发起,由响应方被动响应。为实现各个开放实系统中文件系统开放互连的目的,FTAM 中提出了一个虚拟文件系统概念模型,它屏蔽了实系统中文件系统的实现细节,使得 FTAM 文件服务和协议的定义不依赖于具体的文件系统。图 2 是 FTAM 协议模型。不难看出,FTAM 协议用户服务界面是发起方协议机所提供的文件服务,发起方应用进程不在 FTAM 标准定义范围内,用户可以参照 FTAM 定义,灵活地构造各种发起方应用进程,以满足不同的应用要求。

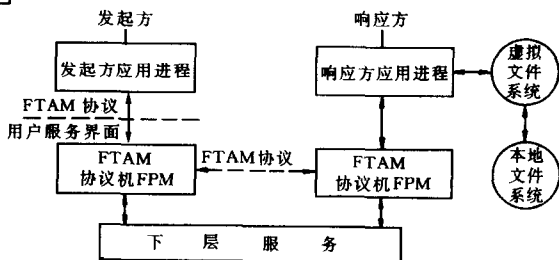


图 2 FTAM 协议模型

FTAM 文件服务定义了广泛的服务原语。按服务质量分类,FTAM 标准定义了两个服务等级,即用户可纠错的文件服务和可靠的文件服务。对于前者,任何差错的出现将由用户处理;对于后者,可恢复的错误将由协议自动纠正,用户不会感到这些错误的发生。

按服务功能分类,FTAM 标准定义了传送、存取、管理、传送和管理、非约束等 5 个服务类,它们由一些功能单元组成。FTAM 标准定义了 9 个功能单元,它们是核心功能单元、读功能单元、写功能单元、文件存取功能单元、有限文件管理功能单元、增强文

件管理功能单元、成组功能单元、恢复功能单元和重启功能单元。

参考文献

1. 曹东启等. 计算机网络软件基础. 北京: 人民邮电出版社, 1982
2. 梁振军等. 计算机互联网技术与 TCP/IP 协议. 北京: 海洋出版社, 1991
3. 胡道元. 计算机局域网(第三版). 北京: 清华大学出版社, 2002 (李学农)

wenjian gongxiang

文件共享 (file sharing) 不同用户共同使用某些文件的技术。文件共享是文件系统所提供的一项重要功能,它最早出现在分时多用户操作系统(如 UNIX 系统)中。文件共享不仅是完成共同的任务所需要的,而且还能节省大量主存空间和辅存空间,减少输入输出操作,更主要的是为用户的应用提供了极大的方便,节省了用户的精力。

随着计算机系统和计算机网络技术的迅速发展,在越来越高的应用需求驱动下,文件共享从作为分时共享模式应用逐步演变为资源共享和客户-服务器模式应用。

在分时共享模式下,共享文件方法通常分为两种:

(1) 由系统来实现对文件的共享 当用户需要共享系统文件(如库文件、标准过程文件等),或共享已知的其他用户的文件及其路径时,由系统从根目录开始为其查寻要共享的文件,并提供相应的操作。

(2) 对需要共享的文件进行联接 如果用户要经常使用其他用户的文件,用上述方法来共享就显得太慢,而且不方便。可以在用户自己的符号文件目录中对需要共享的文件建立相应的表目,这种方法称为联接。联接可以在目录树的结点之间进行,也可以在结点和叶之间进行。但在 UNIX 系统中,联接只能在结点和叶之间进行。值得注意的是联接后的目录结构已经不是树状结构,而成为网状结构,因而路径不是惟一的。尤其在删除目录树的分支时,要考虑到是否有联接,否则,有些目录的联接指针很可能指向一个已被删除的目录表目,从而引起文件访问出错。

资源共享模式以服务器为中心。在这一模式下,局域网把多台 PC 机或 PC 机与专用服务器(如文件服务器、打印服务器等)或多台大型机的资源

集成起来,并提供有效的通信支持,使 PC 机的资源通过局域网得以延伸。文件服务器给用户不仅提供了操作系统中文件系统的一般功能,如生成文件、删除文件等,还提供了文件共享功能。因此,局域网上的 PC 机可以共享局域网中文件服务器上的应用软件和数据库。虽然这种模式能向用户提供灵活的服务,但管理控制和系统维护工具的功能很弱。这种模式是利用 PC 机的能力来运行所有应用,用服务器的能力来作为外设,如硬盘、打印机等的延伸。它的特点为:主要用于共享共同的应用、数据,以及打印机;每个应用提供自己的用户界面,并对界面给予全面的控制;所有的用户查询或命令处理都在 PC 机上完成。

在**客户-服务器模式**下,1 个或多个客户机和 1 个或多个服务器以及下层的操作系统的进程间通信系统,共同组成一个支持分布计算、分析和表示的系统。在该系统中,应用分为前端的客户部分和后端的服务器部分。客户发出请求,网络通信系统将请求的内容传到服务器,服务器根据请求完成预定的操作,然后把结果送回客户。这种模式能充分发挥客户机和服务器的处理能力,从而向用户提供有效的服务,它不只是把服务器当作一个存储数据的大容量外设。

一些流行的网络操作系统是基于客户-服务器模式的操作系统,如 Microsoft 公司的 Windows NT 和 Novell 公司的 NetWare 等。除此之外, Sun Microsystems 公司的网络文件系统 NFS 也是基于客户-服务器模式的。NFS 是在异种机型、异种操作系统以及网络环境中共享文件的网络协议。在一个分布式的网络环境中,NFS 使得一个计算机可以方便地存取其他类型计算机硬磁盘上的数据文件,实现文件共享。

NFS 之所以能在**异构型分布式计算机环境**中透明地提供远程文件存取,与它本身的实现机制有很大关系。NFS 的实现主要是采用远程过程调用(RPC)机制。RPC 提供了一组与机器、操作系统以及下层传送协议无关的存取远程文件的操作。使用 RPC,程序可以如同调用本地过程一样,通过网络调用远程过程。RPC 掩盖了低层的网络细节,为远程服务提供了清晰的、面向过程的接口。

RPC 实现的基础是外部数据表示(XDR)协议。XDR 是一种与机器无关的数据描述编码的协议。它独立于任一机器体系结构的格式,对网上传送的数据进行编码和解码,支持在异构型计算机之间数

据的传送。

为了系统的可靠和用户的安全,文件的共享必须是有控制的。计算机网络系统既应为用户提供共享的便利,又应充分注意到系统和数据(文件)的安全性和保密性。

参考文献

1. 胡道元. 信息网络系统集成技术. 北京: 清华大学出版社, 1996
2. 屠立德. 操作系统基础. 北京: 清华大学出版社, 1990 (李学农)

wenjian guanli chengxu

文件管理程序 (file manager) 操作系统中用于管理和存取数据文件的程序。它采用统一、标准的方法管理在辅助存储器上用户和系统文件数据的存储、检索、更新、共享和保护,并为用户提供一整套操作和使用方法。实现文件管理的程序模块称文件系统。

文件逻辑结构指用户概念中文件数据的排列方法和组织关系。有两类文件逻辑结构:

- (1) 流式结构 相应的文件称流式文件。
- (2) 记录式结构 相应的文件称记录式文件。文件是顺序的若干个逻辑记录的集合。逻辑记录是文件中按数据在逻辑上的独立含义来划分的信息单位。

文件物理结构指文件数据在存储空间中的存放方法和组织关系,有两类方法构造文件物理结构:

- (1) 计算法 设计一个映射算法,通过对记录键的计算转换成逻辑记录的物理地址,从而存取记录。散列文件、顺序文件均属此类。
- (2) 指针法 用指针来表达逻辑记录之间的关系,从而找到逻辑记录的物理地址。索引文件、串联文件、倒排文件均属此类。

文件目录是文件系统实现“按名存取”的主要手段和工具,文件系统的基本功能之一就是文件目录的建立、检索和维护。文件目录应包含有关文件的说明信息、存取控制信息、逻辑和物理结构信息及管理信息。

目录结构可分成:一级目录结构、二级目录结构、树形目录结构。查找目录可用以下方法:顺序查找、二分查找、分级查找和散列查找。

文件共享指一个文件可以让规定的某些用户共同使用。文件共享有两种方式:自动共享和联结共享。

文件保护和保密与文件的共享是互为依存的。文件保护指防止文件拥有者误用或授权者破坏文件;文件保密指不经文件拥有者授权,任何其他用户不得使用文件。两者均涉及用户对文件的访问权限。以下方法可规定使用权限:存取控制矩阵、存取控制表、文件使用权限、文件可访问性。文件保密措施有:隐蔽文件目录、口令、密码。

文件存放在大容量辅存空间中,辅存空间可用以下方法管理:空闲区表、位示图、空闲块链、成组空闲块链等。

文件系统提供一整套操作和使用手段,供用户使用文件。提供的文件类广义指令有:建立、删除、打开、关闭、读、写、控制等;提供的文件类键盘操作命令有:查看目录、改变目录、列目录、建立目录、删除目录、文件改名、文件复制、文件显示、改变文件属性等。

参考文献

- 孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989 (费翔林)

wenyu zhuanhuan

文语转换 (text to speech, TTS) 把文字自动转换成言语(语音)的技术。文语转换(TTS)是言语(语音)合成技术的延伸和扩展。TTS系统中除了语音合成模块外,还包括文本分析、韵律生成模块,如图1所示。

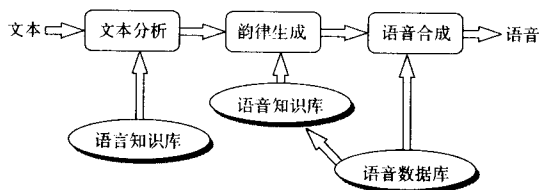


图1 文语转换的处理过程

文本分析是文语转换系统(TTS)的前端。它对输入的文本进行分析理解,为后端语音合成器提供必要的信息,比如读音、停顿、韵律等信息。不同的合成后端需要的信息也各不相同。对于简单的系统,可能文本分析只需要提供读音信息就够了;对于高自然度的任意文本的合成器,文本分析要给出更详尽的语言学或语音学信息,使得合成器合成的语音有更多的可调节的余地。结合自然语言处理和人工智能的研究成果,在充分“理解”文本的基础上,它的输出信息尽可能做到有正确的读音,有轻重缓

急的标记,甚至包含不同的感情风格等。应该说,理想的文本分析器同时就是一个理想的自然语言理解程序。

韵律首先是一个听感知觉的概念。它能帮助听者更好地理解语音所携带的信息。韵律在感知上表现为语音的音高、速度和音量随时间的变化,在声学参数上表现为基频、音段时长和能量随时间的变化。目前,合成语音的质量还欠理想,其根本问题是不能对自然语流中韵律进行有效的模拟。在自然语流中,人们使用语调、节奏和重音等方式来表达说话者的意向和情感,这些韵律特征是自然语流的重要组成部分。韵律生成是指采用各种方法建立韵律模型,并在言语合成时产生韵律参数。

TTS 系统要想取得高质量的声音,必须具备韵律处理和模拟的功能。语调、节奏和重音这些韵律特征是通过超音段特征——音高、音长、音强及频率分布的变化表现出来的。因此,这些超音段特征的修改成为韵律合成的基础。超音段特征的修改可通过多种方法实现,如修改基频模式、共振峰模式等。其中 PSOLA 算法获得了广泛的应用,它为波形合成方法实现 TTS 提供了有力的支持。

要实现韵律模拟,需解决韵律规则、韵律描述、计算模型和修改算法等问题。这要借助于语音学、语言学、心理学、信号处理等学科的成果。首先要研究韵律变化的特点,抽取韵律规则,找出韵律与声学参数的映射关系,给出定量的数学描述,建立计算模型。最后还要设计韵律修改算法。目前对自然语流中韵律现象的研究还远未达到人们的期望。

韵律建模就是要计算言语中的韵律特性,建立韵律模型,以产生这些韵律参数。通常韵律模型分为两类,一种基于规则,另一种基于大规模语料库,采用数据驱动方法。当然它们也不是绝对分开的。

语音合成有着广阔的应用前景。它可应用于盲人计算机、电话信息查询、文本校对、专家系统的有声输出,以及火车站和飞机场的信息报告等领域。

参考文献

1. 蔡莲红,魏华武. 文语转换系统的研究与实现. 应用声学. 1994,13(6)
2. Jan P H van Santen, Richard W Sproat, Joseph P Olive, Julia Hirschberg. Progress in Speech Synthesis. New Jersey: Murray Hill, 1995 (蔡莲红)

wenli shengcheng

纹理生成 (texture generation) 产生物体表面细节的图形生成过程和技术。物体的表面细节称为纹理。纹理是表达物体质感的重要特性,纹理的生成使图形更具有真实感。通常纹理可分为两种不同的类型——颜色纹理和几何纹理。颜色纹理是在物体的光滑表面上描绘附加定义的图案或花纹,它依赖于物体表面的光学属性。纹理可以是规则的或不规则的。规则纹理如墙壁贴面的花纹、平面文字图案等;不规则纹理可以是随机生成的图形或任意的平面扫描图像等。通常所说的颜色纹理,均指二维的平面纹理,定义在二维平面空间。二维的颜色纹理可以看成是一个连续的纹理函数。为了将给定的纹理函数映射到物体表面上就必须进行颜色纹理映射(指颜色纹理的生成技术),这可以通过采用纹理函数值作为物体表面的光亮度参数(如漫反射亮度)用于光照模型的计算来实现。颜色纹理亦可以分布和定义在三维的图形实体内部,称为体纹理。几何纹理或凹凸纹理则是在物体表面上产生凹凸不平的形状,以表达物体表面的微观几何特性。几何纹理通常是由扰动物体表面的法线方向而生成,其方法首先由 James Blinn 于 1978 年提出和实现。纹理生成的过程或技术常称作纹理映射。颜色纹理和几何纹理的生成则分别称作颜色纹理映射和几何纹理映射。

参考文献

唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2001

(吴恩华)

wenli yingshe

纹理映射 (texture mapping) 为了表示物体表面细节,将一个纹理函数映射到物体表面上,并且在绘制该物体表面时,将纹理函数值作为参数来调节表面上各点的光亮度的技术和过程。采用纹理映射的益处在于可以使用较少的处理代价来表现物体表面细节,特别是当纹理函数为二、三维栅格数据时,便于使用图形硬件完成。通常,纹理映射至少包含从纹理空间到物体空间(纹理空间是指纹理函数存在的空间,物体空间是指物体表面存在的空间)的映射及从物体空间到屏幕空间(屏幕空间是指光栅化设备的逻辑坐标系)的映射。

大多数情况下,纹理函数是以二维图像形式描

述的。在此情况下,纹理函数从纹理空间映射到三维物体表面的过程,直观上讲,就是将二维图像“贴”到三维物体表面上。这里首先需要解决的问题是物体表面参数化,即物体表面上任意点 (x, y, z) 都可以表示成 $(x, y, z) = v(u, v)$,其中 v 为一映射, (u, v) 称为纹理坐标。在此基础上,再建立 (u, v) 坐标到图像位置 (i, j) 的关系,即 $(u, v) = p(i, j)$,其中 p 为一映射,通常称为纹理变换。这样,物体表面上任意一点 (x, y, z) 上的纹理值为图像上位置为 $(i, j) = p^{-1}(v^{-1}(x, y, z))$ 的像素颜色所确定。在实践中,纹理映射的困难之处在于物体表面参数化,这里的困难体现在两个方面:①如何对任意拓扑结构任意几何形状物体表面构造如上所述的映射 v 。②若要纹理在物体表面上不发生扭曲变形,应使物体表面的任意微小面元在通过上述映射后,在图像空间保持形状不变。然而,构造这样的映射 v 和映射 p 是相当困难的。

从物体空间到屏幕空间的映射通常发生在绘制过程中。当屏幕空间中某一像素被物体表面上某一面元所填充时,该像素上的纹理值应为该面元上的纹理均值。而在绝大多数的情况下,严格求出一个面元上的纹理均值在计算量上是不可接受的,因此,人们提出了多种近似计算面元上纹理均值的方法,如最近距离法、线性插值法、MipMap法等。

不同的纹理映射技术主要表现在纹理函数及纹理空间到物体空间的映射方式上的不同。按纹理函数的维数来分类,有一维纹理、二维纹理、体纹理、动态纹理等;按照纹理函数的内容来分类,有颜色纹理、凹凸纹理、法向纹理、位移纹理等;按建立纹理空间到物体空间映射的方法来分类,有平面投影法、柱形投影法、球形投影法、立法体投影法、表面参数化法等。

参考文献

1. 彭群生, 鲍虎军, 金小刚. 计算机真实感图形的算法基础. 北京: 科学出版社, 1999
2. Woo M, Neider J, Davis T, Shreiner D. OpenGL Architecture Review Board. OpenGL(R) Programming Guide: The Official Guide to Learning OpenGL, Version 1.2 (3rd Edition)
3. David F Rogers 著. 计算机图形学的算法基础(第2版). 石教英, 彭群生等译. 北京: 机械工业出版社, 2002 (鲍虎军 华炜)

wu fangfa

吴方法 (Wu method) 用计算机证明几何定

理的一种方法。1977年,吴文俊给出了一类平面几何问题的机械化证明方法,并运用这种方法在机器上证明了一大批平面几何定理。这种几何定理的机械化证明方法,国际上称为吴方法。

一个几何定理包含着假设与结论,证明就是从假设到结论的判断过程。选取适当的坐标系,用 x_1, x_2, \dots, x_n 等表示坐标,如果定理的假设可以写成

$$HS \begin{cases} h_1(x_1, \dots, x_n) = 0 \\ \dots\dots \\ h_k(x_1, \dots, x_n) = 0 \end{cases}$$

而结论可以写成

$$G \quad g(x_1, \dots, x_n) = 0$$

其中 h_1, \dots, h_k, g 均为某一域上的多项式,则定理的证明就转化为:对于满足 HS 的任意一组 (x_1^0, \dots, x_n^0) ,判定是否有 $g(x_1^0, \dots, x_n^0) = 0$?从几何上看,条件 HS 的每一组零点代表了一幅几何构图,定理的证明就是判定由 (x_1^0, \dots, x_n^0) 所确定的几何构图是否具有 G 这样的性质?即要计算 $\text{Zero}(h_1, \dots, h_k) \subseteq \text{Zero}(g)$ 是否成立,这里 $\text{Zero}(f_1, \dots, f_l)$ 表示多项式 f_1, \dots, f_l 的公共零点集。

我们仅粗略地看 h_1, \dots, h_k, g 都是 x 的一元多项式的情形,考察多项式组 h_1, \dots, h_k 与多项式 g 的零点集之间的关系。用 $h_k(x)$ 去除 $g(x)$ 得余式 $r_k(x)$,即 $g(x) = q_k(x)h_k(x) + r_k(x)$;用 $h_{k-1}(x)$ 去除 $r_k(x)$ 得余式 $r_{k-1}(x)$;...;用 $h_1(x)$ 去除 $r_2(x)$ 得余式 $r_1(x)$,则 $g(x) = \sum_{i=1}^k q_i(x)h_i(x) + r_1(x)$ 。若 $r_1(x) \equiv 0$,则有 $\text{Zero}(h_1, \dots, h_k) \subseteq \text{Zero}(g)$ 。

对于一般的多元多项式,也有这种类似的“求余”运算,“余式”等概念,尽管一般情况下“余式”未必为零,但这种思想却是吴方法的出发点。吴方法中,通过对多项式集建立偏序关系,“求余”运算和约化,在有限步内将 h_1, \dots, h_k 化为一极简捷的“同解”多项式组 CS (这里的“同解”实际上是 $\text{Zero}(h_1, \dots, h_k) \subseteq \text{Zero}(CS)$), CS 称为 h_1, \dots, h_k 的特征列。从 h_1, \dots, h_k 到 CS 的机械化实现过程称为吴消元法。正如上述对一元多项式的分析那样,根据 $g(x_1, \dots, x_n) = 0$ 能否由方程组 $CS = 0$ 推出,可以判定出定理的正误。

机械化证明原理(吴方法) 设几何定理的假设条件由多项式组 HS 表示,结论由多项式 G 表示。又 HS 的特征列为 $CS; C_1, \dots, C_r$ 。如果 G 对 CS 的余式为零,则在非退化条件下, $G = 0$ 可由 $CS = 0$ 推出,即定理成立。

吴方法已远远超出了定理机器证明的范畴,它作为多项式组的一种机械化处理方法,在国际上受到重视。

参考文献

吴文俊. 几何定理机器证明的基本原理. 北京: 科学出版社, 1984 (孙吉贵)

wuhan yajie

无焊压接 (solderless crimp connection) 采用压接工具或设备, 将导线插入可塑性金属的压线筒中, 在压线筒的压接部位施加压力, 使压线筒和导线在压接部位产生塑性变形, 将金属表面的氧化层破坏, 形成清洁无氧化层的接点, 以达到稳定可靠的电气连接的技术。

早在 1882 年就有人提出压接的专利, 但由于当时没有专用的端子和工具, 未能推广应用, 第二次世界大战期间才得到发展。现在压接端子有两种结构形式。一种是筒状银焊封口型, 如图 1 所示; 另一种是开口型, 如图 2 所示。

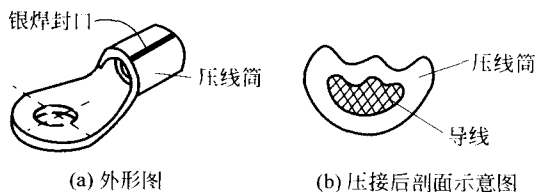


图 1 筒状银焊封口型压接端子

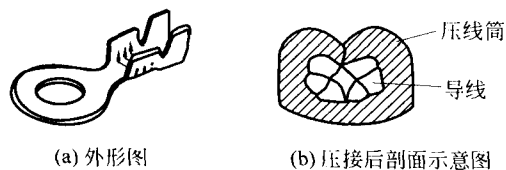


图 2 开口型压接端子

开口型压接片便于做成连锁型端子, 可以卷在圆盘中, 便于压接自动化, 提高了压接的生产效率, 每小时可压上千到几万个连接点, 远高于其他各种连接工艺。

压接工艺简单, 效率高, 最大的优点是导线连接时不用加热, 不需使用助焊剂, 连接后的导线不变硬, 不腐蚀, 已大量地应用在各种连线和连接器中。如美国 AMP 公司能提供 54 000 余种自动、半自动、手动压接工具。压接工艺已被广泛地应用在数据处理设备、户内外供电设备、通信设备、家用电器、汽

车、船舶以及各种军用电子设备的连线和连接器中。

20 世纪 80 年代开始, 计算机采用大规模或超大规模集成电路后, 各单元互连线大量减少, 故机器中电气连接大量采用压接来代替绕接和焊接。

(顾本斗)

wuxian juyuwang

无线局域网 (wireless local area network)

一种灵便的、以无线电波为传输介质的局域网。由于它采用无线电波在空中传送数据。因此能覆盖有线局域网难于涉及的一些场所, 是对有线局域网的必要补充。无线局域网把要发送的数据叠加 (调制) 在无线电载波上, 通过无线电载波传递到接收器, 再由接收器将数据抽取出来。多个无线电载波可以同时存在于同一空间, 只要各自使用的射频 (RF) 不同, 就不会相互干扰。无线局域网与有线局域网相比主要有以下优点:

(1) 流动性 无线局域网允许用户去访问其所机构中任何一处的实时信息。它支持像医院、仓库、机场、停车场等难以布线建网场所的计算机通信活动。

(2) 简便性 无线局域网安装简易, 不需要穿通墙壁和天花板埋设电缆。

(3) 可扩展性 无线局域网能配置成各种拓扑结构去满足不同应用的要求。而且这些拓扑结构易于伸缩, 可从几个用户间的简单连通到成千上万个漫游用户的网络连接。

按采用技术的不同, 无线局域网可分为窄带、扩频、红外和蓝牙技术等。

窄带无线局域网采用一个特定的无线频率去发送和接收用户信息。窄带无线电传输类似于无线电台的广播, 需要将发射器和接收器都调谐到一个“固定”的频率。这种信号可以穿透墙壁, 并扩散到很广的区域, 所以不需要相互瞄准。然而, 窄带无线电也存在一些问题, 例如, 无线电发射、帧检验符 (FCC) 的控制, 以及它们必须精确地调谐, 以防止其他频率的干扰。

扩频技术是一种被军方用于高可靠性和安全性以及至关重要的通信系统的技术, 它消耗比窄带技术更多的带宽, 以便产生更强的信号, 让接收器更易检测到该信号的存在。假若接收器没有调到正确的频率, 这个扩频信号就会被认作背景噪声。扩频无线局域网可以采用跳频扩频 (FHSS) 技术和直接序列扩频 (DSSS) 技术。

(1) FHSS 技术采用一种发送器和接收器双方都知道的,以极精确的频率跳变图形构成一个窄带载波信号来进行发送和接收。经过适当同步后,发送器和接收器间形成一个单独的逻辑信道完成信息传输。对于入侵者而言,只是收到一些短促的脉冲噪声。

(2) DSSS 技术采用发送被称之为碎片的一个有冗余的比特图案去代表要传送的每个比特。这种碎片愈大(即冗余比特大),原始数据能被恢复的可能性就愈大,当然要消耗更多的带宽。甚至在一个碎片的传送过程中,如果有一个或几个比特受损,也可通过适当的统计方法来进行自动校正,不需进行重传。对入侵者而言,其接收器只是收到一些小功率的宽带噪声。

红外(IR)技术在无线局域网方面用的还较少,主要用于规模更小的无线专用网。它能用很高的、接近可见光的频率传送数据,但它像光线一样,也不能穿透墙壁和天花板等不透明物体。

蓝牙(Bluetooth)技术又称 802.15,是一种开放性短距离无线通信技术标准。它面向移动设备间的小范围连接。蓝牙技术是由移动通信公司和移动计算公司联合开发的,传输范围为 10 m 左右,数据传输速率 1 Mb/s,用来在便携式计算机、移动电话和其他的移动设备之间建立一种小型、经济和短距离

的无线链路。新技术性能已达 100 m 左右,数据传输速率约为 10 Mb/s。

蓝牙技术使用 2.4GHz 频段,采用跳频扩频技术,有完全的安全认证技术和灵活的加密方案,支持点对点 and 点对多点的连接。比较适合应用于个人数字助理(PDA)、手机和笔记本电脑等设备。

无线局域网的配置可以简单到仅由几个相互直接访问的无线用户站点所组成,通常称之为 Ad-Hoc(专用)网。Ad-Hoc 网中每个用户站点可以是只装有无线适配卡和天线的个人计算机(图 1)。典型的无线局域网配置除了各个无线用户站点外,还具有一个或多个访问点(AP)。AP 的作用类似有线局域网中的集线器,它通过相应的无线适配卡和天线来连通各用户站点;也可通过电缆与主干网络相连,以便在无线局域网和主干网络间传送和缓冲数据。一个 AP 能支持几百米范围内的一群用户站点。在典型的无线局域网配置(图 1)中,BSS 表示基本服务组,它含有通过同一个 AP 访问主干网络的一群用户站点以及这个 AP。在大型场所,如一个校园,用一个 AP 覆盖不了所有区域时,可以采用几个 AP,这样就形成有多个 BSS。一个用户站点从一个 AP 覆盖区移动到另一个 AP 所覆盖的区域称之为漫游,这是自动完成的,以保证用户可被不间断地连通。

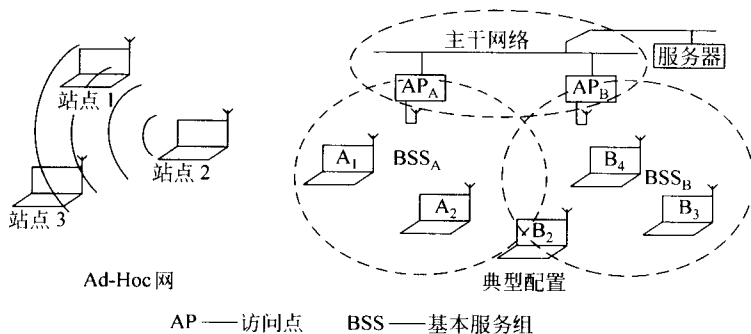


图 1 无线局域网配置

20 世纪 90 年代以来多种无线局域网产品相继推出,为制定其标准,早在 1990 年美国电气和电子工程师学会(IEEE)就成立了 802.11 工作组,但直到 1997 年 7 月才发布了第 1 个标准。这个标准规定的运行频带为 2.400~2.483 GHz,支持 DSSS、FHSS 和 IR 三种技术,数据传送速率为 1 Mb/s 或 2 Mb/s。介质访问控制(MAC)层采用 CSMA/CA(带避免碰撞的载波侦听多址访问)方法。一个站点在发 MAC 帧前,先去侦听介质。如果介质忙碌,就要

推迟这次发送。如果介质空闲,它就再等待一段时间,即一个分配帧间空隙(DIFS),如介质依然空闲就可以发送。接收站接着要去检查所收到的 MAC 帧是否正确(检查 CRC),并发送确认(ACK)帧。假若在一定期间发送站没收到 ACK 帧,它就要重发 MAC 帧。

为延长用户站点供电电池的寿命,IEEE 802.11(参见局域网协议标准)规定两种供电方式,即用户进行发送和接收时的工作方式和不进行收、发时的

省电方式。

802.11 采用通畅信号评价 (CCA) 算法去确定信号是否通畅。这是通过测量天线上的 RF 能量来确定收到信号的强度来实现的。假定强度高于规定的阈值, 这个信号就被称为是畅通的, 可用来传送数据, 否则要按协议规定延缓数据的传输。

随着无线局域网技术逐渐成熟, 在更小范围 (如汽车) 中各种设备的无线连网应用相继出现。这种专供个人使用的设备的范围通常称为私人工作空间 (POS), POS 直径一般小于 10 m, 与 POS 相应的无线局域网也被称之为私人网 (PAN)。IEEE 802.11 工作组从 1998 年 5 月开始就无线私人网 (WPAN) 标准进行研究, 曾定作 IEEE 802.11e 项目。经过近一年的研究, 工作组认为 IEEE 802.11 不适合 WPAN 的要求, 并专门成立了 IEEE 802.15 WPAN 工作组 (1999 年 3 月), 集中精力研究 WPAN 的 MAC 子层和物理层规程标准, 并要求 WPAN 的标准能和 IEEE 802.11 网络协同工作。该工作组经过的一年的工作, 制定出了以蓝牙技术规程为基础的第一个 WPAN 标准。目前采用这种蓝牙技术的产品已涉及世界上半以上的移动电话, 其他像数码相机、笔记本 PC、PCMCIA 卡、便携式游戏系统等也在使用它。

参考文献

1. Pablo Brenner. A Technical Tutorial on the IEEE 802.11 Protocol. 1996. 7. <http://sss-mag.com/pdf/802.11_tut.pdf>
2. Siep T M et al. Paving the Way for PAN Standards: An Overview of the IEEE P 802.15 Working

Group for Wireless Personal Area Networks. IEEE Personal Communication, Feb. 2000

3. 吴企渊. 计算机网络 (第 2 版). 北京: 清华大学出版社, 2004 (黄令恭)

wuxian yingyong xieyi

无线应用协议 (wireless application protocol, WAP) 在 Internet 上所采用的超文本传输协议-超文本标记语言 (HTTP/HTML) 的基础上, 针对移动通信的特点加以修改而形成的通信协议。所谓移动通信的特点即显示界面小、功率低、内存小、中央处理器 (CPU) 运算能力低, 以及带宽低、延迟大和不十分可靠等。HTTP 是专门为万维网设计的一种协议, 在 TCP/IP 体系结构中属于应用层协议 (参见 Internet 体系结构)。HTTP 也是客户-服务器结构 (参见客户-服务器计算), 客户是浏览器, 服务器是万维网 (Web) 服务器。浏览器和服务器通过 HTTP 交换万维网文档。超文本标记语言是一种标记语言, 用来描述如何将文本格式化、通过将标准化的标记命令写在 HTML 文件中, 使得任何万维网浏览器能阅读和格式化任何万维网网页 (参见超文本标记语言)。无线应用协议的诞生是无线应用协议 (WAP) 论坛成员努力的结果。该论坛是在 1997 年 6 月, 由诺基亚、爱立信、摩托罗拉和无线星球等公司共同建立。论坛的目标是通过无线应用协议技术将 Internet 的大量信息及各种各样的业务引入到移动电话及其他无线终端之中。

无线应用协议采用了客户-服务器结构, 提供了一个灵活而强大的编程模型, 如图 1 所示。

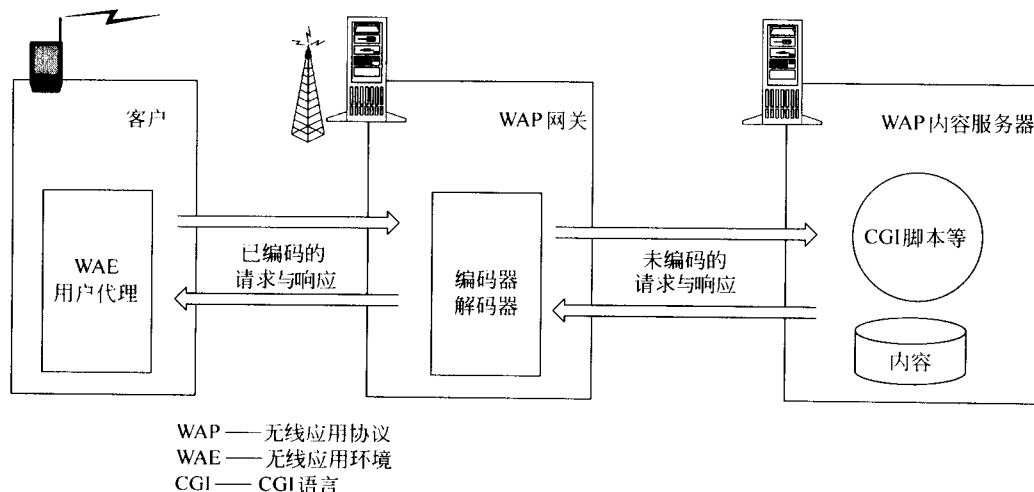


图 1 WAP 编程模型

其中无线应用协议(WAP)网关起着协议的“翻译”作用,是联系移动通信网与 Internet 的桥梁;WAP 内容服务器储存着大量的信息,以提供 WAP 用户来访问、查询、浏览。当用户通过 WAP 终端提出要访问的 WAP 内容服务器的统一资源定位器(URL)后,信号经过无线网络,以 WAP 方式发送请求至 WAP 网关,然后经过“翻译”,再以 HTTP 协议

方式与 WAP 内容服务器交互,最后 WAP 网关将返回的内容压缩、处理成 WAP 客户所能理解的二进制流方式返回到 WAP 客户。编程人员所要做的是编写 WAP 内容服务器上的程序,即 WAP 网页。

如图 2 所示,WAP 定义了一个分层的体系结构,称作 WAP 协议栈,为移动通信设备上的应用开发提供了一个可伸缩的和可扩充的环境。

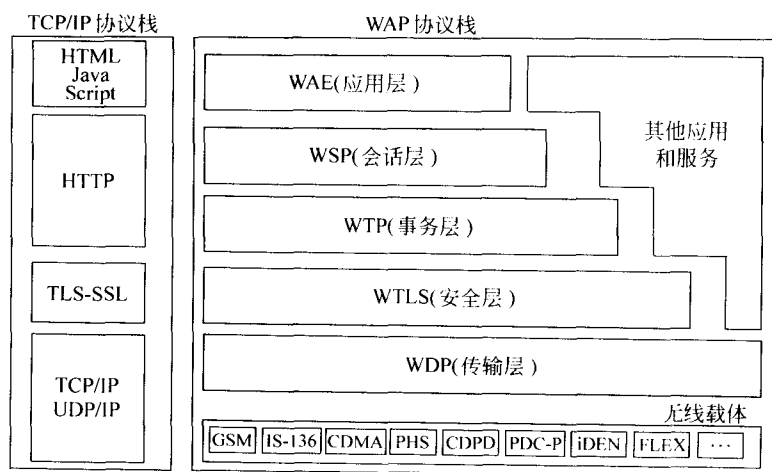


图 2 WAP 协议栈

WAP 协议栈包括了下面几层:

(1) 无线应用环境(WAE)——应用层协议

WAE 是建立在 WWW 技术和移动通信技术相结合的基础上的一个多用途应用环境。它的主要目标就是使得网络系统及内容提供者可通过微浏览器向用户提供不同的内容及应用服务。

(2) 无线会话协议(WSP)——会话协议

WSP 提供两种不同的会话服务:第一种是面向连接的会话服务,运行于事务处理层协议(WTP)之上;第二种是面向非连接的服务,运行于加密或非加密的数据报服务层协议(WDP)之上。WSP 为这两种不同的会话服务向 WAE 提供一致的接口。

(3) 无线事务处理协议(WTP)——事务处理层协议

WTP 运行于数据报服务层之上,提供了一个面向事务处理的轻量级协议,特别适合于小型客户(移动站)。WTP 可有效地运行于加密或非加密的无线数据报网络之上,WTP 提供了三种等级的服务。

(4) 无线传输层的安全协议(WTLS)——安全层协议

WTLS 是工业标准 TLS 协议(即 SSL),用于无线传输的安全协议。WTLS 的目标是使用 WAP 传输层协议,并为在窄带通信信道上使用进行优化。WTLS 具有提供数据的完整性、保密性、验证、拒绝性业务保护等功能。

(5) 无线数据报协议(WDP)——传输层协议

WDP 作为 WAP 的传输层协议,它对上层协议提供一致性服务,并与各种可能的承载服务进行透明通信。因此安全层、会话层和应用层协议都可以各自独立地在 WDP 之上运行。

(6) 无线载体

WAP 可以应用于各种类型的承载服务,包括短消息、电路交换数据和分组交换数据。如果考虑到吞吐量、差错率以及时延,WAP 协议可以容许不同的业务等级,也可以对其进行补偿。WDP 规范说明书列举了它所支持的各种承载业务以及 WAP 承载这些业务所要用到的各种技术。随着无线市场的发展,新的承载业务将会不断地加入。

(7) 其他应用和服务

WAP 的分层结构使得其他服务和应用通过一系列精心定义的接口就可以充分利用 WAP 协议的

功能。外部应用可以直接访问会话层、事务处理层、安全层和传输层。这就使得虽然目前没有被 WAP 所确定,但是对无线市场来说很有市场价值的一些服务和应用也可以使用 WAP 协议。例如,电子邮件、日历、电话号码簿、记事本,以及电子商务、白页、黄页等都有可能使用 WAP 协议。

参考文献

1. WAP forum. Wireless Application Protocol Architecture Specification version 30 - Apr - 1998, <http://www.wapforum.org>

2. 马成清,李娟. WAP 体系结构的组成. 通讯世界,2000,6

(史美林 英春)

wuxiangtu

无向图 (undirected graph) 每条边均是无向边的图(参见图论)。设无向图 $G = \langle V, E, \psi \rangle$, 对任意顶点 $v \in V$, 称与 v 关联的边的数目为顶点 v 的度, 记为 $d_G(v)$ 。 G 的极大连通子图称为 G 的分支。所有顶点的度均为自然数 d 的无向图称为 d 度正则

图。设 $n \in I_+$ (正整数集合), 如果 n 阶简单无向图 G 是 $n-1$ 度正则图, 则称 G 为完全无向图, 记为 K_n 。设 n 阶无向图 G 是 n 阶完全无向图 K_n 的生成子图, 则从 K_n 中删去 G 中的边, 所得到的图称为 G 的补图, 记为 \bar{G} 。包含图中每条边恰好一次的路径, 称为欧拉路径。包含图中每条边恰好一次的回路, 称为欧拉回路。包含图中每个顶点恰好一次的路径, 称为哈密顿路径。包含图中每个顶点恰好一次的回路, 称为哈密顿回路。有欧拉回路的无向图称为欧拉图。有哈密顿回路的无向图称为哈密顿图。无向图的邻接矩阵和路径矩阵的定义与有向图一致(参见有向图)。设无向图 $G = \langle V, E, \psi \rangle$ 无自圈, $V = \{v_1, v_2, \dots, v_n\}$ 且 $E = \{e_1, e_2, \dots, e_m\}$, 则 G 的关联矩阵 $A(G)$ 是一个 $n \times m$ 矩阵 (a_{ij}) , 其中,

$$a_{ij} = \begin{cases} 1, & \text{若 } v_i \text{ 与 } e_j \text{ 关联} \\ 0, & \text{否则} \end{cases}$$

(张强)

X

xitong chengxu sheji yuyan

系统程序设计语言 (systems programming language) 可以用于书写计算机系统程序的语言。

系统软件不同于应用软件,主要用于支持计算机本身的操作和使用,因此它通常和运行该程序的计算机硬件结构有密切的关系,并且往往是常驻的,经常在起作用。因此,对系统程序设计语言有不同于一般高级语言的要求。一般高级语言力求实现与机器无关,以便隐蔽硬件结构,具有良好的易移植性。系统程序设计语言在保持高级语言优点的同时,特别强调程序的结构与功效,必须提供使用简便又兼顾实现功效的系统功能描述手段,有时就不得不包含少量与实现有关的成分。一种良好的系统程序设计语言应当在不牺牲易移植性的条件下实现上述要求。

在 20 世纪 60 年代以前,计算机系统程序都是用汇编语言或机器语言书写的。用低级语言书写系统程序生产率低下,质量控制和维护都很困难。因此,从 50 年代末开始研制能够书写编译程序,特别是能够书写自身的编译程序的高级语言。1958 年出现的 NOLIA 语言具有开创性的意义。最初的系统程序设计语言产生的代码质量不高,曾引起争议。随着语言研制的进展,同时也由于软件规模的日趋大型化使得低级语言已难以满足程序人员的需要,自 70 年代初,高级语言取代了汇编语言,成为系统程序的主要描述工具。这期间出现的 PL 360, BLISS, PASCAL, XCY, Modula-2, Edison 以及 Ada 等语言均可以有效地用于书写系统程序。

目前,系统程序设计语言用于书写各种类型的系统程序,书写 UNIX 系统的 C 语言是使用最广泛的计算机程序设计语言之一。

参考文献

徐家福. 系统程序设计语言. 北京: 科学出版社, 1983

(陈道蓄)

xitong jianrongxing

系统兼容性 (system compatibility) 为一种计算机系统开发的软件或硬件可适用于另一种或其他多种计算机系统的能力。

系统兼容性是系列计算机的基本特性,是在计算机硬件技术迅猛发展,市场竞争剧烈,生产厂商不断推出新产品型号的情况下,避免用户在老产品型号上开发的软件遭受废弃的一种重要设计思想与技术措施。它保护了用户的已有资源,节约了厂商和用户的开发投资,加快了计算机的研制过程,促进了计算机产业和应用的发展。

随着计算机硬件技术的发展,计算机的处理能力不断提高,应用领域急剧扩大,对软件的需求量不断增加,但软件的生产技术发展缓慢,开发效率低,周期长,费用高。如何使一种软件产品能尽量在多种计算机上使用是人们关心的问题,广大用户在要求更新计算机时,希望原有软件能在新的计算机上继续使用。计算机的生产厂家为取得更大的经济效益,更希望以巨资开发研制的软、硬件产品能用于多种计算机上。20 世纪 60 年代由 IBM 公司开发的 IBM 360 系列计算机,可视为实现了系统兼容性的早期代表。在此之后,具有兼容性的系列机大量出现。

兼容性表现在软件和硬件的许多方面,兼容的范围和级别也各不相同,其实现方法分述如下:

(1) 机器语言程序兼容 机器语言就是用硬件实现的机器指令。实现用机器语言编写的程序兼容对计算机体系结构有非常苛刻的要求,需要实现兼容的两台计算机的体系结构和操作系统的用户程序接口等应完全相同;即使略有不同,也可用软件模拟或硬件仿真实现兼容。但这些方法将使用户程序的运算速度明显降低。

(2) 汇编语言程序兼容 汇编语言是一种低级的符号语言,是机器语言的符号化,与机器语言密切相关。因此实现兼容所面临的问题与机器语言基本相同。但用汇编语言编写的程序要经过汇编程序汇编成机器语言程序才能在计算机上运行,这就给实现兼容提供了一些方便。汇编语言程序兼容的首要

条件是要在实现兼容的计算机上配有兼容的汇编语言文本及其汇编程序。如果语言文本不同,可以用转换程序解决,如用户程序与操作系统的接口差异可通过汇编程序转换解决,缺少某种指令可由汇编程序用广义指令来实现其功能等。但如果实现兼容的计算机体系结构差别较大,则汇编语言程序的兼容将难以实现。

(3) 高级语言程序兼容 高级程序设计语言文本与机器的体系结构无关,所以用高级程序设计语言编写的程序容易实现兼容。这是用户程序实现兼容的主要方法。采用标准化的语言文本,则更容易实现兼容。但实际上,由于在不同计算机上对语言文本有不同的限制,可能影响兼容性的实现。

(4) 系统软件兼容 系统软件是控制计算机运行和为用户程序服务的一类软件,主要包括操作系统、编译程序、数据库管理系统等。编译程序将用高级语言编写的源程序编译成机器语言程序,它与计算机的体系结构密切相关,因此,不同体系结构的计算机之间难以实现编译程序的兼容。操作系统是与计算机体系结构密切相关的系统软件,但从功能上看,操作系统可分为与计算机体系结构相关的部分和与用户程序相关的部分。前者构成操作系统的内核,不同计算机的操作系统有不同的内核,互不兼容;后者为操作系统的外层,与计算机体系结构无关,可以实现兼容。数据库管理系统也与计算机体系结构无关,可在操作系统的支持下,实现兼容。

(5) 软件系统兼容 在软件的发展过程中,新的软件系统不断出现,因此也产生了各种软件系统之间的兼容问题。为了使在某种软件系统环境下开发的软件能在新的软件系统环境下正确运行,就需要新开发的软件系统与以前的软件系统兼容,如要求与某操作系统兼容,与某数据库管理系统兼容等。FoxPro 数据库管理系统就是采用向下兼容的技术与dBASE III, FoxBASE 等数据库管理系统兼容。各种计算机上配置的 UNIX 操作系统在外层上也有不同程度的兼容性。

(6) 设备或部件兼容 设备或部件兼容是指一种设备或部件可不加改动地用于多种机器。这要求设备或部件符合某种标准化设计,包括设备或部件的功能、接口、约定、规范、规程等。

(7) 系列机 系列机是计算机体系结构相同、具有标准的外围设备接口,软件向上兼容、性能和价格不同的大、中、小各种型号配套的一族机器。系列机既实现了软件兼容,也实现了硬件兼容,充分体现

了兼容性的实现原则和优越性。

(8) 兼容机 一些计算机厂家为了利用别人的软件成果,研制了兼容机。这些计算机体系结构可能不同,厂家也各不相同,但软件兼容,有的还实现了插件兼容。这种兼容机是选择一种市场前景较好的计算机作为兼容对象,按照这种计算机体系结构设计出可利用其软件资源的计算机。这种兼容机不但体系结构与兼容对象相同,甚至部件也是一样的。

随着计算机软、硬件技术的发展,设计人员更加注意产品的兼容性。实现兼容性的技术和方法会越来越受到人们的重视。但兼容性要求也往往给新产品的开发提出各种制约,降低了计算机体系结构变革的自由度。

(时方缙)

xitong wei hu

系统维护 (system maintenance) 为使计算机系统保持在(或重新恢复到)正常工作状态所需采取的维护措施,包括检查、测试、调整、更换设备和修理等。其目的是为了减少或缩短系统停机时间,提高系统的可用性(参见计算机可用性)。

早期的计算机多采用面板维护方式。在面板上设置若干开关、板键、指示灯等元、器件,由手动方式进行维护。随着计算技术的发展,维护技术也有了很大的提高。目前在较大的系统中一般均设有维护处理机支持系统的调试和维护,较小的系统则通过内置维护诊断固件或使用一些专门的自动检查技术进行维护。计算机硬件故障则多采用诊断技术,由计算机自动实现故障检测和定位。诊断方式可以采用自诊断、互诊断或他诊断模式。

预防性维护与事后处理

系统维护方式通常可分为预防性维护和事后维护两种情况。

预防性维护 又分为定期维护和视情维护两种。

定期维护是以工作时间或储存时间来确定其维修周期,定期检查性能参数是否在规定的容许范围之内,发现那些只有通过测试、检查、保养才可发现的故障和故障隐患,并加以排除,检查备件有无失效以及数量是否符合规定要求,以保证和恢复固有可靠性,预防故障的发生。

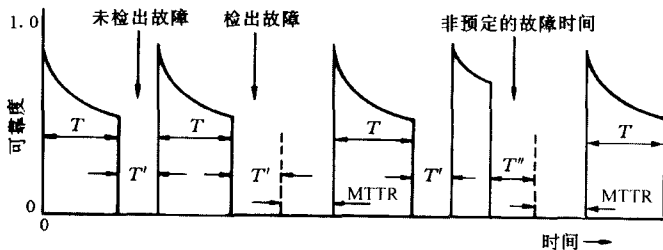
视情维护是根据计算机系统在使用中的技术情况和环境条件的需要来确定维修时机的维修方式,而不规定固定的维修周期。视情维护的针对性强,

可以有效地预防故障的发生,减少维修工作量和早期故障的发生概率。

事后维护 是发生故障后,进行故障定位,采取必要的修理与更换措施,使其恢复到正常工作状态的维修作业。

平均修复时间

计算机在使用过程中要经受检验、运行、故障、修理等周期。一个经维护的系统的周期性能,如图1所示。



T —— 容许的最大“可用”时间 T' —— 由于预防性维护的停机时间
 T'' —— 非预定的诊断 $MTTR$ —— 平均修复时间

图1 一个经维护的系统的周期性能

修复时间是从发现1次失效到恢复至正常状态所需时间,它包含诊断、故障定位、修理实施等时间。平均修复时间 $MTTR$ 是修复时间的平均值(参见计算机 RAS 技术)。

可维护性、可服务性

可维护性是设计与安装的一个特性,是当系统失效后根据规定的过程与资源实行维护时,在一给定的时间间隔内恢复到规定功能的能力。在 t 时间间隔内被修复的概率称为可维护度 M :

$$M = 1 - e^{-t/MTTR}$$

式中 M —— 可维护度;

t —— 指定的修复时间;

$MTTR$ —— 平均修复时间。

为了提高计算机的可用性,不仅要提高系统固有可靠性,而且要认真研究可服务性设计方法。影响可服务性的因素主要有:模块化设计程度、故障检测方法、可达性和可更换性。

可服务设计应包括以下内容:服务性指标的确定与分配、故障诊断设计与故障排除技术、模块化设计、结构的可达性设计、维修体制与预防性维修时间的确定等等。

参考文献

李海泉. 电子计算机系统的故障诊断及维护系

统. 北京: 机械工业出版社, 1984

(杨肃英)

xitong xingneng zhibiao

系统性能指标 (system performance specification) 反映计算机系统性能和特征的一组参数。

不同的用户对系统性能指标的关心不同,例如科学计算的用户最关心 MIPS、MFLOPS、加速比;军事用户最关心可靠性和环境适应性;过程控制人员

最关心实时性和可靠性;维护人员最关心可用性和可维护性。有关可靠性、可用性和可维护性参见计算机系统可靠性、计算机可用性和计算机可维护性。此外,最常用的性能指标还有 MIPS、MFLOPS、加速比、吞吐率、响应时间、利用率、性能价格比等。通常,性能指标可分为绝对的和相对的两类。所谓绝对的是指不需要一定参照量的参数,如 MIPS 和 MFLOPS。所谓相对的是指相对于一定参照量的参数,如加速比是多结点并行处理速度相对于单结点处理速度的倍数。

VAXMIPS 是相对于 VAX 11/780 系统的 MIPS 参数。性能指标又可分为基本的和导出的。所谓基本的是指不能由其他参数导出的参数,这种参数是直接获得的,如处理时间 T 。所谓导出参数是指通过基本参数而间接得到的,如 MIPS, MFLOPS 分别是由指令数和浮点操作数与处理时间导出的;加速比是由多结点处理时间和单结点处理时间导出的。性能指标还可分为工作量类、响应性类和利用率 3 种。吞吐率、工作速度、指令执行速率和数据处理速率属工作量类。响应时间、周转时间和反应时间属响应性类。硬件 (CPU, 内存, I/O 通道, I/O 设备等) 的利用率,操作系统的利用率,公用软件 (如编译程序等) 的利用率,数据库的利用率属利用率类。

MIPS 表示百万 [条] 指令每秒,用来描述计算机的定点运算速度。常用的有峰值 MIPS、基准程序 MIPS 和以特定系统为基准而得到的 MIPS。对于 1 个处理机芯片或 1 台串行计算机,其峰值 MIPS 值通常是以其指令集中最基本指令的执行速度而计算得的。如设某一系统的指令集中最基本指令的执行需 a 个机器周期,每一机器周期为 t 微秒,则其峰值 MIPS 值为 $1/(at)$ 。一台处理机的平均 MIPS 值是指按照一定的加权值平均其指令集中各类指令的执行速度而计算得的 MIPS 值。基准程序 MIPS 值是

指用基准程序测得的 MIPS 值。对于某一计算机,所用基准程序的不同,测得的 MIPS 值也不同,这主要是由于不同的基准程序中各种指令的混合比例不同以及指令集中各种指令的执行速度不同而引起的。MIPS 指标用于评价同一厂商生产的同一系列的计算机的定点运算速度比较准确,因为这些计算机有相似的系统结构、操作系统、语言和编译器,尤其是相似的指令集。如果两台计算机的结构和指令集不同,那么用 MIPS 值来比较它们的运算速度是不准确的,有时可能会导致错误的结论。

MFLOPS 的定义是百万[次]浮点运算每秒,用来描述计算机的浮点运算速度,衡量计算机的科学计算性能。MFLOPS 的定义可分为两种:峰值 MFLOPS 和以基准程序测试得到的 MFLOPS。对于单个处理机芯片或一台串行计算机,其峰值 MFLOPS 通常是以其最快的浮点操作速度或以各浮点操作的平均速度计算得到的。设其最快的浮点操作的执行速度或各浮点操作的平均速度为每个机器周期 a 次,每个机器周期为 b 微秒,则其峰值 MFLOPS 定义为 a/b MFLOPS。串行计算机的 MFLOPS 值也可用基准程序测得。设基准程序 A 在其上执行的时间为 t 微秒,程序 A 中含 F 个浮点操作,则其 MFLOPS 为 F/t MFLOPS。 F 的计算有不同的方法。有的是直接统计机器所执行的浮点操作次数,有的是根据问题的复杂性分析,从数学模型中计算出来。前者获得的是机器实际执行的浮点操作数,其中,可能包括一些辅助性的非问题本身的浮点操作,而后者只包括与问题有关的浮点操作。MFLOPS 可用于比较和评价在同一系统上求解同一问题的不同算法的性能。MFLOPS 还可用于在同一源程序、同一编译器以及相同的优化措施、同样的运行环境下,利用相同的方法对不同系统测试浮点运算速度。由于实际程序中各种操作所占比例不同;不同浮点操作的运算速度不同;对于传统的超级计算机, MFLOPS 值没有考虑向量化程度对运算速度的影响;对于并行处理系统, MFLOPS 值没有考虑诸如通信、竞争共享资源等对运算速度的影响; MFLOPS 值没有考虑运算部件与存储器、I/O 系统、互连网络等速度之间的相互协调等因素。所以 MFLOPS 值只能近似地说明在特定条件下系统的浮点运算速度。

加速比是度量多结点并行处理比单结点处理的加速倍数,用来描述并行处理的效果。绝对加速比以当前解决问题的最好串行算法作为比较基准,它

着眼于并行处理相对于串行处理的优化效果上,用于评价并行算法。相对加速比定义为同一并行算法在单结点上运行时间与多个相同结点构成的并行处理系统上运行时间之比。它着眼于并行算法和计算机本身的可扩展性,用于评价计算机系统的性能。

吞吐率指计算机在单位时间内能处理的信息量。响应时间指从给定计算机输入到出现对应的输出之间的时间间隔。

响应时间取决于用户输入的信息、系统特性以及在用户输入信息时系统正在处理的其他负载。虽然响应时间是一个随机变量,但在相当长的时间周期内可以用统计规律描述它。

利用率指在给定的时间内,计算机某一部分的实际使用时间所占比例。

在事务处理领域中,常用事务处理每秒(TPS)来表示计算机的处理速度,但“事务”没有统一的定义。部分计算机厂商以[次]运算每秒(OPS)表示计算机的处理速度。

性能价格比也是常被用作评价计算机的参数之一。它以每元人民币或每美元所能买到多少 MIPS 或 MFLOPS,或以每 MIPS 或 MFLOPS 值多少人民币或美元来表示。

上述系统性能指标一定程度上反映了计算机系统的性能和特征,但有时不够精确,甚至有时会得出错误的结论。必须注意,上述系统性能指标均与工作负载以及系统特性有关,评价者必须清楚是在什么系统和什么样的负载情况下测得的性能指标,否则,是没有意义的。

(郑纬民)

xitong zongxian

系统总线 (system bus) 在计算机系统中,在多于 2 个模块(部件或子系统)之间传送信息的公共通路。计算机采用系统总线不仅可以大大简化硬件的设计过程,简化系统结构,使系统易于扩充;而且大大简化系统的软件设计过程,减轻软件的设计和调试工作负担,缩短了软、硬件的研制周期,降低了系统的成本。

系统总线又称为**板级总线**或**内总线**,因为它是计算机系统内部各插件板之间传输信息的公共通路。它的位置和用途有别于**片总线**和**外总线**。片总线又称**器件级总线**,它是处理器芯片内部引出的总线。外总线又称**通信总线**,它是计算机系统与计算机系统之间,或是计算机系统与其他系统(仪

器、仪表或控制装置)之间的信息传输通路。

为了在各模块之间实现信息共享和交换,总线由传输信息的信号线及一套管理信息传输的通用规则(协议)所构成。

系统总线的信号线大致可分为五类:①数据传输信号线,包括地址线、数据线以及读-写控制线等;②中断信号线,包括中断请求线、中断认可线等;③总线仲裁信号线,包括总线请求线、总线许可线和总线忙线等;④其他信号线,包括系统时钟、复位、电源和地线等;⑤备用线,用于扩充功能或特殊用途。目前常用的系统总线有 ISA, EISA, VME, MULTIBUS, PCI 等。

系统总线最基本的任务是保证信息在总线上可靠地传输。为了使信息源和信息接收部件能同步,在总线上传输信息时必须遵守一定的定时规则,这种定时规则称为总线定时协议。总线定时协议通常分为3种:①同步总线定时协议,所有模块都连接到公共时钟上,总线操作都在公共时钟控制下的固定时间内进行;②异步总线定时协议,总线操作由信息源或信息接收部件的特定跳变所确定;③半同步总线定时协议,总线操作之间的时间间隔按公共时钟周期的整数倍变化。

系统总线的—个重要的性能指标是它的带宽。总线带宽是总线本身所能达到的最高传输速率,其单位是 MB/s 或 Mb/s。例如, EISA 总线的数据宽度为 32, 工作频率为 6 ~ 8.33 MHz, 其最大带宽为 33.32 MB/s。总线带宽受3种因素的制约:①挂在总线上的模块数量;②总线布线的长度;③总线驱动器和接收器的性能。

随着微电子技术和计算机系统设计技术的迅速发展,系统总线也在不断地发展和完善,从性能和技术上看,目前总线向以下几个方面发展,一个是支持工业控制应用的总线,要求具有强有力的工业输入输出支持,组合灵活,价格低廉,一般只要求支持 8 位和 16 位微处理器,如 STD 总线、STE 总线等。另一个是提高系统处理能力的总线,如要求能支持 32 位和 64 位微处理器,以及支持高速网络传输和多媒体传输等,属于这一类的总线有 VL 总线、PCI 总线、工作站中的 InfiniBand 总线 Futurebus+ 等。还有一类是适应便携式计算机的小型板卡的总线,如 PCMCIA 等。

(孙德文)

xibao zidongji

细胞自动机 (cellular automaton) 并行计算

机的一种理论模型。细胞自动机可视为由若干小单元构成的动态阵列,其中每一单元具有有限个状态,在离散步序中,每一小单元按一致的法则,由其原状态及其邻域单元的状态决定新的状态。在任一时刻,诸小单元状态的总体构成细胞自动机的格局。从初始格局到最后格局的演化过程为计算处理过程,最后格局被视为计算结果。实际上,细胞自动机的每一小单元均为—有限态自动机,细胞自动机即为有限态自动机的动态阵列,细胞自动机格局的演化过程即为并行计算过程。细胞自动机的主要功能在于:可由局部特性及简单的一致性法则模拟、处理总体上具有高复杂性的离散过程和现象。所以,细胞自动机除了在计算机科学中的重要意义外,它还是描述现实世界中大型复杂离散系统的数学工具。

细胞自动机的“小单元”在细胞自动机结构中称为细胞。一个细胞自动机定义为有规律地分布于 n 维空间中的一个细胞空间或细胞集合。 $n=1$ 是最简单情形,细胞分布在一条直线上,每一细胞为一方格,每一细胞有两个邻域细胞(简称邻域),每一细胞仅有两个不同状态,可由 0 和 1 值表示。在每一时刻,全部细胞状态值构成的序列为细胞自动机在该时刻的格局。在 $n=2$ 的情形,细胞在二维平面上分布,它们可为连续的方格、等边三角形、蜂房形式或其他形式。一个简例是,细胞空间为 z^2 , 其中 z 为整数。每一细胞 $x = (x_1, x_2)$ 直接连接它的 8 个邻域,每一细胞的状态值 $q \in \{0, 1\}$ 。局部性法则为局部传递函数 δ :

$$\delta(q_0, q_1, q_2, \dots, q_8) = \begin{cases} 1, & \text{若 } q_0 = 0 \text{ 且 } \sum_{i=1}^8 q_i = 3 \\ & \text{或 } q_0 = 1 \text{ 且 } 2 \leq \sum_{i=1}^8 q_i \leq 3, \\ 0 & \text{其他情形} \end{cases}$$

其中 q_0 为所考虑的细胞的状态。可知,细胞在离散步序中的新值由该细胞的当前值及其 8 个邻域细胞的当前值按法则 δ 确定。在每一时间步序, δ 以并行方式同步地作用于每一细胞状态,从而实现细胞自动机总体格局的并行计算。一般地,细胞自动机的基本模型具有五个主要特征:

- (1) 它们由细胞的离散格局构成;
- (2) 它们在离散时间步序中演化;
- (3) 每一细胞的状态均在同一有限集中取值;

(4) 每一细胞的状态依同一确定的法则演化;

(5) 细胞状态的取值法则仅依赖于其自身及其周围局部领域细胞的状态值。

由于不同的目的和需要,出现了许多细胞自动机的变异模型。这些变异由以下因素产生:细胞相互联接形式的改变;细胞取值随机性的引入;细胞间信息交换(通信)机制;外部信息的输入及信息输出,等等。这使得在文献中也使用了其他一些术语。

细胞自动机提供了分布并行计算系统的数学模型,它在新一代计算机结构设计中具有重要意义。细胞自动机在模式识别、图像处理及人工智能中有重要应用,在计算机视觉研究中是合宜的模型。细胞自动机的理论研究,涉及计算理论中众多深层次问题。

实际上,细胞自动机的理论和应用不仅限于计算机科学本身。由于细胞自动机可描述具有很大自由度的离散动态系统,它可视为偏微分方程离散化的理想形式。所以,它在物理学中有广泛应用,特别是对非线性问题的处理。细胞自动机的另一些重要应用是在生物学中,涉及到生命进化过程、神经网络等方面。作为其逻辑结构和功能可被细胞自动机阐明的一个生物系统,D. Farmer 等人给出了研究 DNA 序列的一个新方法,指出,DNA 序列的演化,遗传信息的存储和更新,细胞自动机作为模型是适宜的。

在现实世界中,自然现象的许多个别或局部规律人们是易于了解的,但构成现象的整体特性往往非常复杂,难以把握。细胞自动机提供了研究这类现象的一种手段。一个简单例子是细胞自动机可模拟雪花生成的过程和机制。对细胞自动机格局演化过程中极限集的研究,可发现分数维问题和混沌现象。细胞自动机是研究混沌理论的数学工具之一。

细胞自动机理论和应用的研究不过短短四十余年,处于开始阶段。伴随计算机科学的发展,它的作用将会日益显现。

参考文献

1. Farmer D, Toffoli T, Wolfram S. Cellular Automata. Amsterdam: North-Holland, 1984
2. Demongeot J, Golès E, Tchuente M. Dynamical Systems and Cellular Automata. London: Academic Press, 1985
3. Toffoli T, Margolus N. Cellular Automata Machines. Cambridge: MIT Press, 1987 (周广福)

xiatui zidongji

下推自动机 (pushdown automaton) 带有栈的有限自动机。该栈表是一个存储量没有限制的辅助存储器,是一个“先进后出”的栈。存入一个符号(压栈)就把从前放入栈里的符号顺次往下推一次,取出一个符号(弹出)必定是取栈顶符号,栈内其他符号顺次向栈顶移动一次。取放重叠于一弹簧上的盘子就是弹出压栈的一个形象表示。下推自动机主要讨论由上下文无关文法产生的上下文无关语言的关系,下推自动机的类型和子类,程序语言语法的确定型下推自动机,下推转换器和不可判定问题等。

下推自动机各种变形及其子类 下推自动机(PDA)的数学定义为七元组 $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, 其中有有限集 Q 为状态集,有限集 Σ 为输入字母表集, Γ 为栈符号集, $q_0 \in Q$ 称为初态, $Z_0 \in \Gamma$ 是一个称为栈开始符号的特殊符号, $F \subseteq Q$ 称为终态集, $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow Q \times \Gamma^*$ 是多值函数。例如

$$\delta(q, \varepsilon, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \dots, (p_m, \gamma_m)\}$$

表示 PDA 在状态 q 时与扫描的输入字母无关,栈顶符号为 Z ,可以转移到状态 p_1, p_2, \dots 或 p_m ,并以 $\gamma_1, \gamma_2, \dots$ 或 γ_m 相应于 p_1, p_2, \dots 或 p_m 代替 Z ,且输入头不动。定义表明,下推自动机是一个不确定的机器。 $L(M) = \{x \in \Sigma^* \mid \delta(q_0, x, Z_0) = (p, \varepsilon, \gamma), p \in F, \gamma \in \Gamma^*\}$ 称为下推自动机按终态接收的语言。研究处理这类语言的算法时,必须准备好从前面错误选择转移动作里恢复正确选择的办法,但这往往是很麻烦的,为了避免这种情况,引入下推自动机按空栈接收的语言,它的定义为 $N(M) = \{x \in \Sigma^* \mid \delta(q_0, x, Z_0) = (p, \varepsilon, \varepsilon), p \in Q\}$ 。已经证明下推自动机 M_1 按终态接收的语言必为某一下推自动机 M_2 按空栈接收的语言,且任一 M_2 按空栈接收的语言必为某一 M_1 按终态接收的语言,即 $L(M_1) = N(M_2)$, PDA M_1 与 M_2 等价。PDA 与形式语言理论中乔姆斯基分层的上下文无关语言等价,即任一上下文无关语言必有一个 PDA 接收它,且 PDA 接收的语言必为上下文无关语言。任一上下文无关语言状态数最多可以为 2,无 ε -转移的按终态的 PDA 所接收,也可以为 $\delta(q, a, Z)$ 的栈符号不超过 2 的 PDA 所接收。满足条件:对每一 $q \in Q$,每一 $Z \in \Gamma$, (1) 当 $\delta(q, \varepsilon, Z) \neq \emptyset$ 时,对所有 $a \in \Sigma, \delta(q, a, Z) = \emptyset$ 。(2) 对所有 $a \in \Sigma \cup \{\varepsilon\}, \delta(q, a, Z)$ 的元数不多于 1 的 PDA 称为确定下推自动机(DPDA)。条件(1)说明与输入无关的转移和与输入有关的转移不会产生混淆,避

免了扫描一个输入字母和出现使用空字的自发转移的不确定性。条件(2)说明 δ 为单值函数或单值偏函数,从而表明了确定性。若 DPDA 的栈操作只有擦去栈顶符号或压入一个栈符号,则称该 DPDA 为标准形。任一 DPDA 存在一个等价的 DPDA,它扫描任一输入字的全体字母。带 ε -转移的 DPDA 和不带 ε -转移的 DPDA 的功能不同。又,DPDA 类是 PDA 的真子类,如语言 $\{a_1 a_2 \cdots a_n a_n \cdots a_2 a_1 \mid a_i \in \Sigma\}$ 可以作为一个 PDA 所接收,但不能被任何 DPDA 所接收。允许输入头在输入带的两个方向移动的 PDA 称为双向下推自动机(2PDA),它接收在终态时移动离开右端的输入串。语言 $L = \{0^n 1^n \mid n \geq 1\}$ 可被一个 2PDA 所接收,但不能被任何 PDA 所接收,故 PDA 与 2PDA 不等价。 L 是某一 DPDA 所接收的语言,如果 $x \in L$, x 的任何真前缀都不在 L 中,称 L 具有前缀性质。已证明具有前缀性质的上下文无关语言 L 可由某一 LR(0) 文法(参见 LR(k) 文法)所生成。为了给许多程序语言提供方便而自然的语法分析生成器,增加朝前看一个符号的 DPDA(对应于 LR(1) 文法)就能识别更多的语言,因而 DPDA 对编译设计极为重要。

下推自动机的运算,预测机和不可判定问题

PDA 经替换、同态映射、逆同态映射后仍为 PDA。PDA 与有限自动机之交、PDA 对有限自动机的商,求 INIT、CYCLE、reversal 仍为 PDA。但两个 PDA 之交、PDA 之补不再是 PDA。DPDAM = $(Q_M, \Sigma, \Gamma, \delta_M, q_0, Z_0, F_M)$ 与有限自动机 $A = (Q_A, \Sigma, \delta_A, q_0', F_A)$ 产生的机器 $\Pi(M, A) = (Q_M, \Sigma, \Gamma \times Q_M \times Q_A, \delta, q_0, x_0, F_A)$ 称为预测机,其中 $\delta: Q_M \times \Sigma \times (\Gamma \times Q_M \times Q_A) \rightarrow Q \times \Gamma^*$, $x_0 = [Z_0, \mu_0]$, 且 δ 和 μ_0 满足若干附加条件。用预测机可以得到 DPDA 若干运算性质,如 DPDA 在补、MIN、MAX 运算下仍为 DPDA, DPDA 与有限自动机之交、对有限自动机的商仍为 DPDA,但在同态映射下,或经并、连接、克林闭包运算后就不再是 DPDA 了。PDA 是不是一个 DPDA,两个 DPDA 的交是否为空,两个 DPDA 之交、并是不是 DPDA,一个 DPDA 是不是另一 DPDA 的子自动机等都是不可判定的。但有算法判断一个 PDA 所接收的语言是否为空、是否为有限,且有 CYK 算法判断一个字是否为一个给定的 PDA 所接收。

下推转换器 数学定义为 $M = (Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F)$, 其中有限集 Q 是状态集,有限集 Σ 是输入字母集, Γ 为栈符号集,有限集 Δ 为输出字母集, q_0 为初态, $F \subseteq Q$ 为终态集, $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow$

$Q \times \Gamma^* \times \Delta^*$ 是转移函数。机器从 q_0 出发,输出带为空,栈顶符号为 Z_0 ,输入串 x 在输入带上,机器运行时,从左到右逐次扫描 x 的每个字母, δ 根据当前状态 p ,当前输入带上的内容 a ,当前栈顶符号 Z 规定出允许的结果 (q, y, γ) , 其中 q 为下一状态, y 为输出带上面下一段的输出字,栈顶的符号串 γ 。下推转换器与下推自动机的运行相似,但另外生成一个输出串作为处理的一部分。 M 用终态方式生成的翻译集合 $T(M) = \{(x, y) \mid x \in \Sigma^*, y \in \Delta^*, y \text{ 在输出带上}, M \text{ 从 } q_0, Z_0 \text{ 开始,扫描完 } x, M \text{ 在 } F \text{ 的一个状态时停止运行}\}$ 。用空栈方式生成的翻译集为 $T_\varepsilon(M) = \{(x, y) \mid x \in \Sigma^*, y \in \Delta^*, y \text{ 在输出带上,栈空时停止运行,接收 } x\}$ 。已经证明,用空栈方式生成的翻译集当且仅当它是上下文无关句法制导翻译模式(SDTS)系统的翻译集。由 SDTS 构造的下推转换器直接用于模式识别,如描述亚中央染色体和远端染色体串的染色体文法。下推自动机模拟可接收输入串的最左派生(导出),是分析上下文无关语言自顶到底分析器的标准模型,当出现可供选择的分支时,下推变换器总是选择正确动作,这个假定在数学上是虚构的,但是这种虚构使得不必逆向跟踪或并行推导,使问题简化。对下推变换器引入算子(将 Σ 的元素变为其他字母的错误)可以校正翻译中的错误,并已推广到概率自动机的研究中。

参考文献

1. Hopcroft J E and Ullman J D. Introduction to Automata Theory, Languages, and Computation. Addison - Wesley, Reading, Mass; 1979
2. Gonzalez R C, Thomason M G 著. 句法模式识别. 濮群,徐凤家,徐光佑译. 北京:清华大学出版社,1984

(张一立)

xianshiqu

显示器(display device) 由监视器、显示适配器(显示卡)和有关电路组成的用以显示数据、图形、图像的设备。是计算机系统中重要的输出设备。向计算机输入的各种信息、计算机运算和处理的结果,都通过显示器转换成肉眼可见的字符、图形和图像。

显示器的种类 显示器种类繁多,可以按不同的标准来分类。

按显示器件分 ①发光器件。如阴极射线管(CRT)、等离子体(PDP)、发光二极管(LED)、电致

发光管(ELD)等,其中CRT显示器是最常用的。这些器件在外加电信号后本身会发光。②光调制器件。如液晶显示(LCD)、电化学反应显示(ECD)等。这些器件本身不发光,但在电信号作用下,其光学特性起变化,使光线透过或反射,组成人眼可见的图像。LCD显示器已广泛应用于电子手表、计算器、笔记本计算机和台式计算机等产品上。

按所用显示适配器分 有MGA显示器、CGA显示器、VGA显示器、SVGA显示器、多频显示器等。

按屏幕尺寸分 常见的有35 cm(14英寸)、43 cm(17英寸)、50 cm(21英寸)等。

按显示颜色分 ①单色显示器。只能显示两种颜色。可显示的颜色有白色、黑色、橘红色、琥珀色、绿色等。这类显示器体积小、重量轻、价廉,适用于流动性较强的场合。②彩色显示器。能够显示许多种颜色。

按输入信号形式分 ①输入为数字信号。这类显示器的输入信号是分离式的TTL脉冲信号,MGA及CGA显示器属于这一类。②输入为模拟信号。这类显示器输入信号有3个模拟信号,理论上它可显示无穷多种彩色,但实际的彩色种类还受显示适配器的控制。VGA及SVGA显示器属于这一类。

显示器的基本术语和主要技术指标

亮度 亮度是指荧光屏发亮的等级。亮度可分为4级,即暗、淡、亮和特亮。画面的亮度和显示点的发光强度、发光时间两者大体成正比,在有些图形显示要求更亮时,可通过延长发光时间来提高其亮度。一般在室内较亮的环境下,显示器亮度应大于 120 cd/m^2 (坎每平方米)。

对比度 对比度是指荧光屏画面上最大亮度与最小亮度之比。一般显示器应大于300:1。

屏幕尺寸 习惯上用屏幕对角线的长度表示。

像素 屏幕上能够独立控制其颜色和亮度的最小区域。计算机对图像信息的处理是以像素为基本单位的。

点距 点距是指显像管荧光屏上相邻两个像素之间的距离。在彩色显像管中也就是相邻的同一种色点之间的距离。常见的点距有0.31 mm, 0.28 mm, 0.25 mm, 0.21 mm等几种,点距数值越小图像越清晰。

分辨率 是图像清晰程度的标志。通常用屏幕上一行和一列所包含像素的个数的乘积来表示,即由水平方向的像素个数和垂直方向的像素个数的乘积来表示。如 640×480 ,表示水平方向的像素个数

是640个,即有640列;垂直方向的像素个数是480个,即有480行。常见的分辨率有 1024×768 , 1024×1024 , 1600×1280 , 2048×1536 等。

扫描 CRT显示器件中电子束在显示屏上按某种轨迹运动的过程。CRT显示器件的发光是由电子束打在屏幕的荧光粉上引起的。电子束过后,发光很快衰减而后消失。为了能看到稳定的图像,就必须不断重复扫描。通常电子束在屏幕上运动是从左到右、从上到下。沿水平方向的扫描称为行扫描,沿垂直方向的扫描称为场扫描,相应的扫描频率分别称为行扫描频率(行频)和场扫描频率(场频)。

CRT显示器扫描图像的方法有两种。一种是将一幅画面一行一行地依次扫描,称作逐行扫描。CRT显示器多用逐行扫描。另一种是将一幅画面分成两半,由奇数行组成奇数场,偶数行组成偶数场。先扫奇数场,再扫偶数场。两画面镶嵌在一起产生的视觉还是一个完整的画面,这种扫描方法称作隔行扫描。标准电视制式采用的是隔行扫描。

场频 文字或图形每秒钟在屏幕上重复出现的次数,也就是电子束从左上角到右下角的扫描速度。扫描速度越高,图像越稳定。当场频大于24 Hz时,人眼就有连续感,场频低于47 Hz会有明显的闪烁感,场频高于75 Hz才不会有闪烁感。CRT显示器的场频在60~180 Hz之间。

行频 电子束每秒钟的水平扫描次数,也就是从左到右的扫描速度。显示器的场频和行频有两类:①固定频率的显示器只有一种场频和行频。VGA显示卡的频率必须与显示器一致才能使用。②多频自动跟踪显示器的场频和行频可在一定范围内变化,能支持多种VGA卡。这是常见的类型。对于逐行扫描,

$$\text{行频} = 1.1 \times \text{场频} \times \text{垂直分辨率}$$

CRT显示器的行频在30~120 kHz之间。

视频带宽 指视频放大电路可以处理的频率范围。在CRT显示器中,视频带宽由分辨率和场扫描频率决定。

彩色阴极射线管(CRT)显示器 彩色显示器的工作是基于三基色原理。自然界中的各种颜色几乎都可以由相互独立的三种单色光以适当的比例混合得到,这三种颜色被称为三基色。其中任一种基色不能由另外两种基色混合而得到。通常选用红、绿、蓝三种颜色作为三基色。应用三基色原理,可先把彩色图像分解成红、绿、蓝三种基色图像,然后再分别转换成电信号,传送到接收端后再混合成原来

的彩色图像。

CRT 监视器 由阴极射线管、视频放大电路、光点定位电路(行扫描电路、场扫描电路、同步电路)和电源等部分组成。

阴极射线管 又称显像管,是将电信号转变为可见图像的电真空器件。可分为黑白显像管和彩色显像管两大类。彩色显像管有产生红、绿、蓝三种基色的荧光屏和激励荧光屏的三个电子束。由三基色荧光粉所产生的分量不同的光来形成各种颜色。光点定位方法广泛应用的是光栅扫描。电子束在荧光屏上从左到右、从上到下一行行地有规律地扫描,形成一组光栅。根据需要显示的信息对电子束进行调制,形成图形或有明暗层次的图像。

CRT 显示器反应速度快、分辨率高,是应用较多、技术比较成熟的显示器。今后将朝着提高分辨率、提高亮度、增大屏幕、屏幕平面化、降低辐射和使用方便的方向发展。由于它体积大、笨重、要用高电压、有辐射等缺点,在许多应用场合被 LCD 显示器所取代。

CRT 显示适配器 连接计算机的总线与 CRT 监视器以实现监视器控制的接口部件。1981 年,美国 IBM 公司为其个人计算机提供了两种显示适配器。一种是单色显示适配器(MDA),它只有文字显示模式,每屏可显示 80×25 个字符。另一种是彩色图形适配器(CGA),它可以显示字符和以点绘制的图形。1985 年,IBM 公司提供了较高分辨率的增强型彩色图形适配器(EGA)和视频图形阵列适配器(VGA),VGA 最高分辨率达 640×480 ,可同时显示 16 种颜色,得到了广泛的应用。后来,各种高性能的产品不断出现,显示速度、分辨率和位深度都有很大提高。

显示适配器通常组装成电路板,直接插在微型计算机总线上使用,故又称**显示卡**。

显示适配器一方面通过主机板上的扩展槽与主机系统总线连接;另一方面通过多芯电缆将视频信号、亮度信号、垂直和水平同步信号等送往显示器。它由寄存器、显示存储器、只读存储器、显示处理器和接口电路等部分组成。移位寄存器把输出数据变成串行数据,经视频电路转换成视频信号。显示存储器保存要显示的图像数据,即屏幕上的每一个像素的亮度和颜色信息。每个像素占用的位数越多,则能表示的色彩种类和亮度的层次也越多。例如每个像素只用一位,不是 1 就是 0,像素也只有两种颜色,不是黑就是白。用 8 位表示,有 $2^8 = 256$ 种颜色;用 16 位表示,有 $2^{16} = 65\,536$ 种(通常称为**高彩**

色);而用 24 位表示,则有 $2^{24} = 1.67 \times 10^7$ 种颜色(称为**真彩色**)。通过改变显示存储器中的内容,就可改变屏幕上显示的图像。

显示适配器有不同的型号和规格,必须与相应规格的监视器配套使用。

液晶显示器(LCD) 一种本身不发光的显示器件。通过对环境光的反射或对外加光源控制的方式来显示图像。

液晶发现于 1888 年,它在一定温度范围内既有晶体所特有的各向异性,又具有液体所特有的流动性。利用它在电场作用下能沿电场的方向排列成行,透光率随电压改变的特性,1968 年 RCA 公司制成了**液晶显示器**。1973 年,夏普公司把它用于电子表和计算器。

液晶显示器种类 从物理效应上分有动态散射型和扭曲向列型。动态散射型多用于仪器、仪表和计算器,扭曲向列型多用于计算机显示器。从结构上可分为笔段型和点阵型。笔段型多用于仪表,点阵型用于计算机显示器。

液晶显示器的基本结构 由两片刻有透明导电电极的平板玻璃基板夹着一个液晶层组成,但实际结构要复杂得多。

液晶本身不发光,它需要一个亮度高且均匀的背光源。彩色液晶显示器上使用最广泛的背光源是冷阴极荧光灯(CCFL)。在最底层和灯在一起的还有导光板和增亮膜,前者把线光源雾化成均匀的面光源,后者把光线聚拢使其垂直进入液晶层。在光源上面是玻璃基板,基板上面有偏振膜和驱动矩阵薄膜晶体管(TFT)电路。然后紧贴着的是液晶层,在液晶层上面,除了有玻璃基板、偏振膜和驱动矩阵薄膜晶体管电路之外,还有滤色片。

具有扭曲向列效应的液晶在未加电场时,其分子排列平行于电极表面,但在上表面与下表面之间旋转 90° 。这种旋光特性在外加电场作用下会减弱或消失。如果液晶层上下表面的偏振膜的光轴是平行的,那么不加电场时由于液晶的扭曲效应,光线通不过偏振膜,因而像素对应的点是暗的。加电场后液晶不发生扭曲,光线通过偏振膜,像素对应的点是亮的。如果偏振膜的光轴是互相垂直的,不加电场时点是亮的;加电场后点是暗的。每个像素的红、蓝、绿三色各有一个。驱动矩阵电路中的薄膜晶体管。当薄膜晶体管通导时,有电场加到液晶上,对光线进行调制。RGB 滤色片把可见光分解成三原色,进而组成各种颜色以得到彩色画面。

液晶显示器的特点 ①液晶显示器的工作电压低,功耗小。这是其他任何显示器件无法比拟的。②液晶显示器件的结构便于利用集成电路工艺进行批量生产。③无辐射。液晶显示器件不用高电压,在使用时不会产生 X 射线和电磁波辐射。④每个像素都有三个独立的晶体管,允许对每个像素直接寻址,并有记忆的功能。不需要扫描,也就没有闪烁和图像畸变。⑤重量轻、厚度薄、寿命长。

和 CRT 显示器比较,液晶显示器也有不足之处:①响应速度慢。液晶分子扭转过程需要较长的时间,当图像快速运动时会出现拖尾和模糊现象。2003 年响应速度大多数只达到 25 ms,少数达到 16 ms。②亮度差。受到外加电源所用荧光灯品种、发热、功率等限制。③对比度差。④可视角度小。

液晶显示器的控制与 CRT 显示器不同,要用相应配套的液晶适配器。

液晶显示器在电子表、仪器仪表和笔记本电脑等领域中已广泛应用。随着性能的提高和成本的降低,在台式计算机中的应用也越来越多。

参考文献

周怡聪主编. 多媒体计算机外部设备. 北京:清华大学出版社,2002 (林兼)

xianchang zongxian kongzhi xitong

现场总线控制系统 (fieldbus control system, FCS) 基于现场总线的分布式控制系统。它以现场总线为纽带,将挂载在总线上的单个分散的测量、控制设备作为网络结点连接成网络,并与受控对象、传感器、执行机构一起构成自动控制系统,实现对生产过程的监视、控制和对现场仪表工作状态的监视、参数修改的功能。

这里的现场总线是指应用在生产现场,在微机化测量控制设备之间实现双向串行多结点数字通信的底层局部控制网络,可广泛应用于制造业、流程工业、交通、楼宇等方面的自动控制系统。

系统组成 在系统组成上,现场总线控制系统与集散控制系统(DCS)在系统构成、功能、控制策略等方面有一些类似之处,如图 1 所示。现场总线控制系统由现场总线仪表、现场总线网络设备、现场总线传输介质、组态与监控计算机以及符合现场总线协议的组态软件、监控软件、设备管理软件、嵌入式万维网服务器等组成。

现场总线仪表是指具有现场总线接口的现场仪

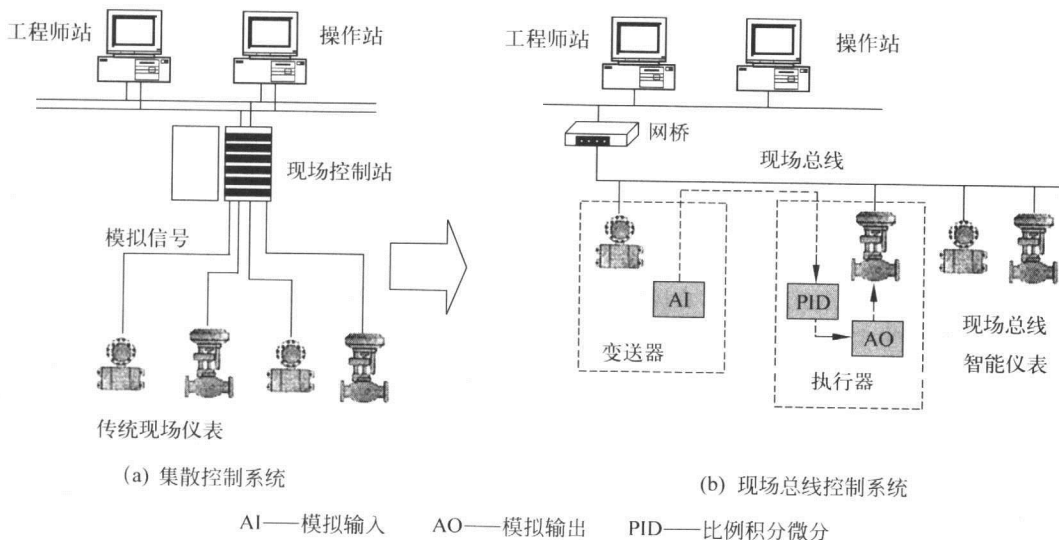


图 1 集散控制系统与现场总线控制系统比较

表(如检测仪表、分析仪表、执行仪表等)和二次仪表(如显示仪表、记录仪表以及控制装置等),这些现场总线仪表除了实现传统的测量、执行功能外,还具有现场总线通信接口,可以包含执行控制功能所

需的一个或多个软件模块——功能块(FB),如模拟输入(AI)、模拟输出(AO)、比例积分微分(PID)、数字输入(DI)、数字输出(DO)等。

现场总线网络设备是指现场总线连接器、中继

器、终端器、网桥、不同现场总线之间的协议转换器或网关、现场总线安全栅等。

现场总线介质则是指现场总线通信中实际传送信息的载体,如**双绞线电缆**(带屏蔽或无屏蔽)、**同轴电缆**、**光缆**等。

组态与监控计算机(或称为工程师站、操作站)用于对现场总线设备进行组态和对生产过程进行监控,其中操作站只能用于对生产过程监控,不能用于组态。

组态软件的作用是,对现场总线控制系统的网络参数(如地址、端口、网络角色等)、设备参数(如设备完成特定功能所需的各种参数)以及控制策略(由分布于不同设备内的多个控制功能块组成完整的控制方案)进行组态(参见**集散控制系统**)。

设备管理软件所实现的功能是利用获得的现场设备自身信息(如制造商、设备类型等)、诊断信息以及管理信息等,对现场总线设备进行组态管理、故障诊断、远程调试与标定以及对现场总线设备数据库进行自动维护等。

特点和优点 现场总线控制系统由集散控制系统脱胎发展而来,它保留了 DCS 的精华,除了能实现传统 DCS 的全部功能外,还有以下的优点:

(1) **全数字通信** 从现场变送器开始到整个系统采用完全的数字信号传输方式,使信号的检错、纠错机制得以实现,抗干扰能力和稳定性更高,传输精度显著提高,并可大大节约布线电缆,工程安装周期缩短,维护与系统扩展方便。

(2) **现场设备状态可控** 除了测量、控制信息外,其他非控制信息,如设备类型、型号、厂商信息、量程、设备运行状态等都可以通过现场总线传输到网络上的任何智能设备。操作人员在控制室里就可以对设备进行诊断与预防性维护;对现场设备进行远程标定,并能够及时识别失效设备,以免错误的控制动作发生,又提高了系统的可靠性和可维护性。

(3) **互操作性和互换性** 现场总线是开放的协议,不同厂商生产的符合同一现场总线协议的设备可以连接在一起,统一组态和协同工作;来自不同厂家的相同类型和现场总线协议的设备可以互换。

(4) **系统的开放性** 现场总线标准实现了完全开放,无专利许可要求,面向世界上任何一个制造商和用户。不同制造商生产的设备之间可实现完全的信息交换。

(5) **控制分散** 通过现场总线,将传统集散控制系统(DCS)、可编程逻辑控制器(PLC)等控制系统复杂的控制任务进行分解,分散于现场设备中,由现场变送器或执行机构构成控制回路,并行实现各部分的控制。简化了系统结构,提高了系统的可靠性、自治性和灵活性。

现场总线标准化 现场总线国际标准从 1984 年起开始起草,但由于技术、经济等方面的原因,于 1999 年底通过的现场总线国际标准 IEC 61158 和 2002 年的修订版(第三版)没有能够统一市场上的各种现场总线,而是包含多种类型的总线标准,例如 Foundation Fieldbus、Profibus、ControlNet、P-Net 等。

现场总线控制系统的发展方向 由于以太网具有应用广泛,成本低廉,通信速率高,软硬件资源丰富,可持续发展潜力大等优势,因此它不仅垄断了应用于办公自动化等商用计算机之间的通信,在工业企业的控制层、管理层也得到了广泛的应用,并已成为工业控制领域的主要通信标准之一。

以太网的通信不确定性曾是其应用于工业现场的主要障碍之一。但随着以太网交换技术、全双工通信技术的发展(参见**异步传输**),尤其是通信速率的快速提高(如**千兆位以太网**)使得碰撞已可以避免或大大减少,以太网通信的不确定性已经不再成为应用于工业控制的障碍。

但以太网应用于工业控制现场时,还需解决互操作性问题,以及总线供电、本质安全、远距离传输、网络可用性、网络安全性等关键技术。

国际上一些组织,如工业自动化开放网络联合会(IAONA)、工业以太网协会、IDA 小组等都在研究工业以太网技术,并将其应用于工业控制系统。

参考文献

1. 阳宪惠. 现场总线技术及其应用. 北京: 清华大学出版社, 1999
2. 张浩, 谭克勤, 朱守云. 现场总线与工业以太网络应用技术手册(第一册). 上海: 上海科学技术出版社, 2002 (冯冬芹)

xiancheng

线程(thread) 并发程序(参见**进程**)中共享地址空间的并发执行单位。每个线程是一个并发执行单位,它拥有为执行计算所必须的程序代码及为程序运行所拥有的存储地址空间。一个**中央处理器**可以被多个线程并发地共享,而多个线程共享同一个

其中系数 a_{ij} 为实数或复数, 方程组(1)的矩阵向量形式为

$$\mathbf{A}^{(1)} \mathbf{x} = \mathbf{b}^{(1)} \quad (2)$$

其中 $\mathbf{A}^{(1)} = (a_{ij}^{(1)})$, $\mathbf{b}^{(1)} = (a_{1n+1}^{(1)}, a_{2n+1}^{(1)}, \dots, a_{nn+1}^{(1)})^T$, $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$, $\mathbf{A}^{(1)}$ 与 $\mathbf{b}^{(1)}$ 为已知, \mathbf{x} 为待求。

高斯消去法由消元过程和回代过程组成。消元过程是依次利用前面方程消去后面方程中的未知数, 将方程组(1)变为等价的三角形方程组, 然后通过回代过程求三角形方程组的解。

第1步: 设 $a_{11}^{(1)} \neq 0$, 将(1)式中第1个方程分别乘以 $-a_{21}^{(1)}/a_{11}^{(1)}$, $-a_{31}^{(1)}/a_{11}^{(1)}$, \dots , $-a_{n1}^{(1)}/a_{11}^{(1)}$ 加到第2至第 n 个方程上, 消去这些方程中的 x_1 , 得到与(1)等价方程组

$$\left. \begin{aligned} a_{11}^{(1)} x_1 + a_{12}^{(1)} x_2 + \dots + a_{1n}^{(1)} x_n &= a_{1n+1}^{(1)} \\ a_{22}^{(2)} x_2 + \dots + a_{2n}^{(2)} x_n &= a_{2n+1}^{(2)} \\ \dots\dots\dots \\ a_{n2}^{(2)} x_2 + \dots + a_{nn}^{(2)} x_n &= a_{nn+1}^{(2)} \end{aligned} \right\} \quad (3)$$

其中, 后 $n-1$ 个方程的系数计算公式为

$$l_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}}, \quad a_{ij}^{(2)} = a_{ij}^{(1)} - l_{i1} a_{1j}^{(1)} \quad (4)$$

$$i = 2, \dots, n; \quad j = 2, \dots, n+1$$

方程组(3)的矩阵向量形式为

$$\mathbf{A}^{(2)} \mathbf{x} = \mathbf{b}^{(2)} \quad (5)$$

一般, 设已得到第 $k-1$ 次消去后的结果

$$\mathbf{A}^{(k)} \mathbf{x} = \mathbf{b}^{(k)} \quad (5)$$

则第 k 步消元过程如下: 设 $a_{kk}^{(k)} \neq 0$, 将式(5)中第 k 个方程分别乘以 $-a_{k+1,k}^{(k)}/a_{kk}^{(k)}$, $-a_{k+2,k}^{(k)}/a_{kk}^{(k)}$, \dots , $-a_{nk}^{(k)}/a_{kk}^{(k)}$ 加到式(5)中第 $k+1$ 至第 n 个方程上, 消去这些方程中的 x_k , 得到与式(5)等价方程组

$$\mathbf{A}^{(k+1)} \mathbf{x} = \mathbf{b}^{(k+1)} \quad (6)$$

其中, 后 $n-k$ 个方程的系数计算公式为

$$l_{ik} = a_{ik}^{(k)}/a_{kk}^{(k)}, \quad a_{ij}^{(k+1)} = a_{ij}^{(k)} - l_{ik} a_{kj}^{(k)},$$

$$i = k+1, \dots, n; \quad j = k+1, \dots, n+1$$

如果依次 $a_{kk}^{(k)} \neq 0, k = 1, 2, \dots, n-1$, 上述过程重复执行 $n-1$ 次后得到与(1)等价的三角形方程组

$$\mathbf{A}^{(n)} \mathbf{x} = \mathbf{b}^{(n)} \quad (7)$$

具体形式是

$$\left. \begin{aligned} a_{11}^{(1)} x_1 + a_{12}^{(1)} x_2 + \dots + a_{1n}^{(1)} x_n &= a_{1n+1}^{(1)} \\ a_{22}^{(2)} x_2 + \dots + a_{2n}^{(2)} x_n &= a_{2n+1}^{(2)} \\ \dots\dots\dots \\ a_{nn}^{(n)} x_n &= a_{nn+1}^{(n)} \end{aligned} \right\} \quad (8)$$

以上对 k 的递推过程, 称为消元过程。如果 $a_{nn}^{(n)} \neq 0$, 则由式(8)得出向后递推公式

$$\left. \begin{aligned} x_n &= a_{nn}^{(n)} / a_{nn}^{(n)} \\ x_i &= (a_{in}^{(i)} - \sum_{j=i+1}^n a_{ij}^{(i)} x_j) / a_{ii}^{(i)}, \\ i &= n-1, n-2, \dots, 1 \end{aligned} \right\} \quad (9)$$

称为回代过程。

用高斯消去法求解 n 阶方程组时, 其乘除法和加减法的运算量约为 $\frac{1}{3}n^3 + O(n^2)$ 。

在消元过程中有可能遇到某个元素 $a_{kk}^{(k)} = 0$ 的情况, 但是如果方程组(1)有惟一解, 则在 $a_{k+1,k}^{(k)}, a_{k+2,k}^{(k)}, \dots, a_{nk}^{(k)}$ 中至少有一个不为零, 比如 $a_{k+j,k}^{(k)} \neq 0$, 那么将第 k 个方程与 $k+j$ 个方程对调, 便可继续进行消去。有时虽然 $a_{kk}^{(k)} \neq 0$, 但其绝对值很小, 以它作除数也会引起较大误差, 从而使最后的解不精确, 甚至面目全非。为了避免这种情况, 应将 $|a_{k+1,k}^{(k)}|, |a_{k+2,k}^{(k)}|, \dots, |a_{nk}^{(k)}|$ 中最大者所在方程与第 k 个方程对调。这样的消去过程称为列主元消去法。

三角分解法 该法是直接从方程组(2)出发, 将系数矩阵 \mathbf{A} 分解为两个三角形矩阵之积, 即 $\mathbf{A} = \mathbf{L}\mathbf{U}$ 。当 \mathbf{L} 为单位下三角形矩阵, \mathbf{U} 为上三角形矩阵时, 这种分解称为杜利特尔分解法; 当 \mathbf{L} 为下三角形矩阵, \mathbf{U} 为单位上三角形矩阵时, 称为克劳特分解法; 当 \mathbf{A} 为对称正定矩阵时, 这时 $\mathbf{U} = \mathbf{L}^T$, 这种分解法称为楚列斯基分解法或对称分解法。以杜利特尔和楚列斯基分解法为例说明。在杜利特尔分解中, 矩阵 \mathbf{L} 和 \mathbf{U} 的元素 l_{ij} 和 u_{ij} 由下列递推公式计算。对 $k = 1, 2, \dots, n$

$$u_{kj} = a_{kj} - \sum_{m=1}^{k-1} l_{km} u_{mj}, \quad j = k, k+1, \dots, n$$

$$l_{ik} = (a_{ik} - \sum_{m=1}^{k-1} l_{im} u_{mk}) / u_{kk}, \quad i = k+1, k+2, \dots, n \quad (10)$$

其中令 $\sum_{m=1}^0$ 为零。公式(10)的计算顺序是: 先计算 \mathbf{U} 的第1行, 再计算 \mathbf{L} 的第1列, 然后计算 \mathbf{U} 的第2行, 再计算 \mathbf{L} 的第2列, 如此继续下去, 直到得到 \mathbf{L} 和 \mathbf{U} 为止。此时, (2) 可写为两个方程组

$$\mathbf{L}\mathbf{y} = \mathbf{b}, \quad \mathbf{U}\mathbf{x} = \mathbf{y} \quad (11)$$

再分别解这两个方程组, 得到(2)的近似解。

为避免除数 u_{kk} 为零或其绝对值过小而带来误差, 同样可在得到 \mathbf{U} 的元素 $u_{kk}, u_{k+1,k+1}, \dots, u_{nn}$ 后, 在

它们中选主元,然后进行列对调,而 x 分量也进行相应对调。

在楚列斯基分解中,下三角形矩阵 L 及其转置矩阵 L^T 的元素 l_{ik} 由下列公式计算。对 $k = 1, 2, \dots, n$

$$l_{kk} = (a_{kk} - \sum_{m=1}^{k-1} l_{km}^2)^{1/2} \quad (12)$$

$$l_{ik} = (a_{ik} - \sum_{m=1}^{k-1} l_{im} l_{km}) / l_{kk}, \quad i = k+1, \dots, n$$

由于 A 对称正定, l_{kk} 均不为零,故不必选主元。但在式 (12) 中有开方运算,工作量大,可采用改进的楚列斯基分解法。

迭代法

考虑线性代数方程组

$$Ax = b \quad (13)$$

其中 A 为非奇异矩阵。迭代法的基本思想是:从某个初值向量 $x^{(0)}$ 出发,构造一个向量序列 $\{x^{(k)}\}$,使其收敛于某个极限向量 x^* ,且 x^* 就是方程组 (13) 的准确解。

迭代法分点迭代法和块迭代法两种。点迭代法是指每次从已有的近似解分量求出一个新的近似解分量,如此逐个的求下去,直到求出全部分量为止。然后,重复此过程。块迭代法则是先将系数矩阵 A 、解向量 x 和自由项 b 进行分块,然后将每一子块视为一个元素,并按点迭代法的类似公式进行迭代。下面介绍几种点迭代法。

雅可比迭代法 对于一般的线性方程组 (13), 设 $a_{ii} (i = 1, 2, \dots, n)$ 均不为零,从第 i 个方程中解出 $x_i (i = 1, 2, \dots, n)$ 。(13) 的雅可比迭代格式为

$$x_i^{(k+1)} = \sum_{j=1}^n \frac{a_{ij}}{a_{ii}} x_j^{(k)} + \frac{b_i}{a_{ii}}, \quad (14)$$

$$i = 1, 2, \dots, n; \quad k = 1, 2, \dots$$

若将系数矩阵 A 分裂为下列形式

$$A = D + L + U$$

其中 D 为对角矩阵, L 和 U 分别为对角元素为零的下三角和上三角矩阵。用 $D + L + U$ 代替矩阵 A , 则式 (14) 的矩阵向量形式为

$$x^{(k+1)} = -D^{-1}(L + U)x^{(k)} + D^{-1}b, \quad (15)$$

$$k = 1, 2, \dots$$

赛德尔迭代法 在迭代格式 (14) 中,将第 i 个方程迭代解出的 $x_i^{(k+1)}$ 值代替其后各方程中的 $x_i^{(k)}$, 从而得到赛德尔迭代格式

$$x_i^{(k+1)} = -\frac{1}{a_{ii}} \left(\sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} + \sum_{j=i+1}^n a_{ij} x_j^{(k)} - b_i \right)$$

$$i = 1, 2, \dots, n; k = 1, 2, \dots \quad (16)$$

(16) 式的矩阵向量形式为

$$x^{(k+1)} = -D^{-1}(Lx^{(k+1)} + Ux^{(k)}) + D^{-1}b, k = 1, 2, \dots \quad (17)$$

松弛法 该法分同时松弛法和逐次超松弛法两种。它是利用系数矩阵正定性来加快迭代过程的收敛速度。同时松弛法(也称 JOR 法)迭代格式为

$$x_i^{(k+1)} = x_i^{(k)} - \frac{\omega}{a_{ii}} \left(\sum_{j=1}^n a_{ij} x_j^{(k)} - b_i \right) \quad (18)$$

令 $B = D - A$, 则式 (18) 的矩阵向量形式为

$$Dx^{(k+1)} = (1 - \omega)Dx^{(k)} + \omega Bx^{(k)} + \omega b \quad (19)$$

其中 ω 为松弛因子。当 $\omega > 1$ ($\omega < 1$) 时称为超松弛法(低松弛法)。

逐次超松弛法(也称 SOR 法),它的迭代格式为

$$x_i^{(k+1)} = x_i^{(k)} - \frac{\omega}{a_{ii}} \left(\sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} + \sum_{j=i}^n a_{ij} x_j^{(k)} - b_i \right) \quad (20)$$

此式的矩阵向量形式为

$$(D + \omega L)x^{(k+1)} = (1 - \omega)Dx^{(k)} - \omega Ux^{(k)} + \omega b \quad (21)$$

式(19)和式(21)统一写为

$$x^{(k+1)} = Gx^{(k)} + F \quad (22)$$

其中 G 称为迭代矩阵。对应于(19)和(21)式分别有: $G = D^{-1}[(1 - \omega)D + \omega B]$ 和 $G = (D + \omega L)^{-1} \times [(1 - \omega)D - \omega U]$, F 分别为 $\omega D^{-1}b$ 和 $(D + \omega L)^{-1}\omega b$ 。

松弛因子 ω 的选取是很重要的。冯康等编的《数值计算方法》中给出了最优松弛因子的理论计算公式。实际计算中,往往是在 ω 所在范围内选取若干值试迭代,从中选取使收敛速度快的 ω 值。

下面给出三个迭代法收敛定理。

定理 1 若方程组 (12) 的系数矩阵 A 为严格对角占优矩阵,则 A 为非奇异矩阵,且对任意初值 $x^{(0)}$, 方程组 (12) 的雅可比迭代法和赛德尔迭代法收敛。

定理 2 松弛法迭代格式 (22) 收敛的充要条件为 $\rho(G) < 1$, 其中 $\rho(G)$ 为矩阵 G 的谱半径。

定理 3 若 A 为对称正定矩阵, 且 $0 < \omega < 2$, 则逐次松弛法收敛。

除上述方法外,解线性代数方程组还有共轭梯度法、蒙特卡罗法、解三对角方程组的追赶法等。对于一些特殊线性方程组,如由偏微分方程离散化得到的线性方程组,可用对称逐次超松弛法和隐式交替方向法等迭代方法求解。

关于线性方程组求解,目前国际上提供了著名软件库 LINPACK 可供用户使用。

参考文献

1. 冯康等. 数值计算方法. 北京: 国防工业出版社, 1978
2. Wilkinson J H and Reinsch C. Handbook of Automatic Computation Linear Algebra 2. Berlin: Springer Verlag, 1971 (李晓明)

xianxing luoji

线性逻辑 (linear logic) 由法国数理逻辑学家、计算机科学家 J. Y. Girard 于 1987 年建立的一种非经典的逻辑系统。它源于 G. Gentzen 提出的矢列演算,其特点是推理规则作用于公式序列上,而不仅是单个公式上。此外,它将推理规则明确区分为逻辑推理规则及结构性推理规则。在 4 条结构性规则中,切割规则: $\frac{A \vdash C, B \quad A', C \vdash B'}{A, A' \vdash B, B'}$, 其中 A, A' 等表示公式的序列,在证明中可以消去,这就是著名的甘岑定理。减弱规则,例如 $\frac{A \vdash B}{A \vdash B, C}$; 以及收缩规则,

例如 $\frac{C, C, A \vdash B}{C, A \vdash B}$ 的使用,意味着证明的前提在证明中可使用无穷多次。因而经典逻辑是不可构造的。另一条交换规则: $\frac{A \vdash B, C, D, B'}{A \vdash B, D, C, B'}$ 对于系统的构造性是无害的。线性逻辑与经典逻辑的区别在于前者去除了破坏构造性的两条结构性规则。但其他逻辑联结词的规则需要调整和修正。例如在经典逻辑

中,以下两条规则 $\frac{A \vdash C, B \quad A' \vdash D, B'}{A, A' \vdash C \wedge D, B}$ 以及 $\frac{A \vdash C, B \quad A' \vdash D, B'}{A, A' \vdash C \vee D, B}$ 在系统中包含所有结构性规则的情况下是等价的。而在线性逻辑中就需要引进不同的合取联结词:乘合取 \otimes 及加合取 $\&$ 来表述上面的两条规则。对偶地,需要引进乘析取 \ast 以及加析取 \oplus 。关于线性否定 \perp 的定义如下:

每个原子公式都有两种形式 A 及 A^\perp 。 A 的否定为 A^\perp , A^\perp 的否定为 A , 即 $(A^\perp)^\perp = A$ 。

用德·摩根定律定义带联结词的复合公式的线性否定:

$$(A \otimes B)^\perp = A^\perp \ast B^\perp, (A \& B)^\perp = A^\perp \oplus B^\perp, \\ (A \ast B)^\perp = A^\perp \otimes B^\perp, (A \oplus B)^\perp = A^\perp \& B^\perp.$$

线性否定本身并不是联结词,例如, $(A \otimes B^\perp)^\perp$ 仅仅是 $A^\perp \ast B$ 的另一种表示,后者还可记为线性蕴含式 $A \multimap B$ 。

为简单起见,线性逻辑的序列演算使用单边序列 $\vdash A_1^\perp, \dots, A_n^\perp, B_1, \dots, B_m$ 代替甘岑系统中的双边序列 $A_1, \dots, A_n \vdash B_1, \dots, B_m$ 。

线性逻辑系统的常项符号有 $\perp, 1, \top, 0$ 。

公理及推理规则可表述如下:恒同公理 $\vdash A^\perp, A$; 加公理 $\vdash T, A$; 乘公理 $\vdash 1$ 。切割规则 $\frac{\vdash C, A \quad \vdash C^\perp, B}{\vdash A, B}$ (Cut); 交换规则 $\frac{\vdash A, C, D, B}{\vdash A, D, C, B}$ (\times)。

逻辑规则表述如下:关于加的合取及析取

$$\frac{\vdash A, C \quad \vdash B, C}{\vdash A \& B, C} (\&); \quad \frac{\vdash C, A}{\vdash C \oplus D, A} (1 \oplus); \\ \frac{\vdash D, A}{\vdash C \oplus D, A} (2 \oplus).$$

关于乘的合取及析取

$$\frac{\vdash C, A \quad \vdash D, B}{\vdash C \otimes D, A, B} (\otimes); \quad \frac{\vdash C, D, A}{\vdash C \ast D, A} (\ast); \\ \frac{\vdash A}{\vdash \perp, A} (\perp).$$

也可加入对偶的模态联结词 $!$ (A 当然 A) 和 $? A$ (为何不是 A) 扩充线性逻辑系统。其否定为 $(! A)^\perp = ? A^\perp, (? A)^\perp = ! A^\perp$ 。

关于模态词的推理规则为

$$\frac{\vdash B, ? A}{\vdash ! B, ? A} (!); \quad \frac{\vdash A}{\vdash ? B, A} (?); \\ \frac{\vdash ? B, ? B, A}{\vdash ? B, A} (C?); \quad \frac{\vdash B, A}{\vdash ? B, A} (D?).$$

系统中同样可引进谓词,但线性逻辑的主要特征已表现在命题演算部分中。

线性逻辑具有两种含义完全不同的语义。Tarski 传统的语义只注重逻辑演算的结果,而与证明过程无关。线性逻辑的 Tarski 传统语义称为相空间语义,由可换幺半群 $(P, \cdot, 1)$ 以及一个取定的 P 的子集 \perp 构成,记为 $(P, \cdot, 1, \perp)$, 将线性逻辑中的常项、公式解释为相空间中的特定子集,将逻辑符号解释为相空间中的运算。如果一个公式 A 在相空间 P 中的解释为 \bar{A} , 且 $1 \in \bar{A}$, 就称 A 在 P 中有效。线性逻辑演算 (不包含 $!$ 及 $?$) 相对于其在相空间中的有效性是可靠且完全的。另一种更重要的语义是由 Girard 重新开拓的 Heyting 传统的语义。其目的是建立证明的模型。例如假定原子公式的证明已知,

公式 $A \wedge B$ 的证明为由 A 的证明 p 及 B 的证明 q 构成的序对 (p, q) ; $A \rightarrow B$ 的证明是一个函数 f , 它将 A 的每一个证明 p 映射到 B 的证明 $f(p)$ 。线性逻辑的紧连空间语义就实现了 Heyting 的上述思想, 首先将每个命题及常项对应到紧连空间, 由联结词构造出的每个公式也归纳地对应到相应的紧连空间。例如 X, Y 分别为公式 A, B 对应的紧连空间, 则 $X \multimap Y$ 定义为由 X 至 Y 的全体线性函数的轨迹构成的紧连空间, 此紧连空间对应公式 $A \multimap B$ 。我们用相同的符号表示 A 式及它所对应的紧连空间。在这种语义下可以证明: 若 π 是线性逻辑中 $\vdash A$ 的一个证明, 则存在紧连空间 A 中惟一的对象 π^*, π^* 描述了证明 π 。此定理说明, 一个公式的可证性是由公式的内在结构完全决定的。

线性逻辑的另一重要结果是建立了证明的范式, 称为证明网络。证明网络可看作是线性逻辑的自然推理系统。由于线性蕴含 $A \multimap B = A^\perp \star B$, 证明网络中就可避免使用经典逻辑中的蕴含引入和消去规则, 具有十分规范的形式。

线性逻辑对于构造性问题的成功研究为证明论及计算机科学理论中提出的一些基本问题开辟了新的途径。

参考文献

1. Girard J Y. Linear Logic. Theoretical Computer Science, 1987(50)
2. Girard J Y, Lafont Y, Taylor P. Proofs and Types. Cambridge University Press, 1989 (黄且圆)

xianxing wenfa

线性文法 (linear grammar) 一种受限的上下文无关文法。如果上下文无关文法 $G = (\Sigma, V, S, P)$, P 中的所有生成式为形式 $A \rightarrow u$ 或者 $A \rightarrow uBv$, $A, B \in V, u, v \in \Sigma^*$, 则称 G 为线性文法。线性文法生成的语言称为线性语言。例如 $G_1 = (\{a, b\}, \{S\}, S, P_1)$, 其中 $P_1 = \{S \rightarrow aSb, S \rightarrow ab\}$, 是线性文法, 生成的线性语言 $L(G_1) = \{a^n b^n \mid n \geq 1\}$ 。正则语言类是线性语言类的真子类, 而线性语言类是上下文无关语言类的真子类。

超线性文法 线性文法的一种扩展。设上下文无关文法 $G = (\Sigma, V, S, P)$, 如果存在有限多个两两无交的(可能为空的)变量集合 V_0, V_1, \dots, V_n , 使得 $V = \bigcup_{i=0}^n V_i$ 且对于每个 V_i 及任意的 $A \in V_i$, P 中的所有生成式为形式 $A \rightarrow \alpha$, $\alpha \in (\Sigma \cup V_0 \cup V_1 \cup \dots \cup$

$V_{i-1})^*$, 或者 $A \rightarrow uBv$, $B \in V_i$ 且 $u, v \in \Sigma^*$, 则称 G 为超线性文法。特别地, $n = 0$ 时, G 为线性文法。超线性文法生成的语言称为超线性语言。例如 $G_2 = (\{a, b\}, V, S, P_2)$, 其中 $V = \{S, A, B\}$, $P_2 = \{S \rightarrow AB, A \rightarrow aAb, A \rightarrow ab, B \rightarrow aBb, B \rightarrow ab\}$ 。取 $V_1 = \{S\}$, $V_0 = \{A, B\}$, 则 $V = V_0 \cup V_1$, 且 P_2 中的生成式均为超线性文法要求的形式, 所以 G_2 是超线性文法, 生成的语言 $L(G_2) = \{a^n b^n, a^m b^m \mid n \geq 1, m \geq 1\}$ 是超线性语言。

有限转向的下推自动机 S. Ginsburg 等人于 1965 年提出的下推自动机的一种限制类型。以空存储接受的下推自动机 $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \emptyset)$ 在接受一个字的过程中, 栈符号由 Z_0 开始, 栈符号个数时增时减, 最后排空。栈符号个数由增加到减少或者由减少到增加均称为一次转向。如果下推自动机 M 接受的每一个字转向次数均不超过 $2k - 1$, k 为某正整数, 则称 M 是 $2k - 1$ 次转向的下推自动机。有限转向的下推自动机 M 接受的语言用 $\text{Null}(M)$ 表示。例如 $M = (\{q\}, \{a, b\}, \{Z_0, A, B, a, b\}, \delta, q, Z_0, \emptyset)$, 其中 $\delta(q, a, a) = \{(q, \lambda)\}$, $\delta(q, b, b) = \{(q, \lambda)\}$, $\delta(q, \lambda, Z_0) = \{(q, AB)\}$, $\delta(q, a, A) = \{(q, Ab), (q, b)\}$, $\delta(q, a, B) = \{(q, Bb), (q, b)\}$, 是一个 3 次转向的下推自动机, 它接受的语言 $\text{Null}(M) = \{a^n b^n, a^m b^m \mid n \geq 1, m \geq 1\}$ 。对于任意的有限转向的下推自动机 M , 可以构造一个超线性文法 G , 使得 $L(G) = \text{Null}(M)$ 。反之亦然。从而有限转向的下推自动机接受的语言类和超线性语言类是同一语言类。特别地, 1 次转向的下推自动机接受的语言类等同于线性语言类。

参考文献

- Harrison M A. Introduction to Formal Language Theory. Massachusetts: Addison - Wesley, 1978
(郭清泉)

xianxing youjie zidongji

线性有界自动机 (linear bounded automaton) 输入带上以特殊符号 $\$$ 分别作左、右端标识, 读头只在 $\$$ 和 $\$$ 之间移动, 且在 $\$$ 处不能打印任何其他符号的非确定图灵机。

$M = (Q, \Sigma, \Gamma, \delta, q_0, \$, \$, F)$ 称为非确定线性有界自动机(简记为 LBA), 其中 Q 为状态的有限集合, Γ 为带上允许符号的有限集合, $\Sigma \subseteq \Gamma$, Σ 为输入符号集, $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ 为转移函数, L 表

示左移一格, R 表示右移一格, $q_0 \in Q$, q_0 称为初始状态, $\varnothing, \$ \in \Sigma$, F 为终止状态集, $F \subseteq Q$. M 所接受的语言 $L(M) = \{x \mid x \in (\Sigma - \{\varnothing, \$\})^*, \text{且 } q_0 \varnothing x \$ \xrightarrow{M} \alpha q \beta, q \in F, \alpha, \beta \in \Sigma^*\}$ (\xrightarrow{M} 意指: 从 q_0 经有限次转移函数作用于字 x 可到达 q), 称为上下文有关语言或 1 型语言, 记为 CSL. 若上述定义中采用确定型图灵机, 则称 M 为确定型线性有界自动机 (DLBA), 并称 $L(M)$ 为确定的上下文有关语言 (DCSL). CSL 类是递归可枚举 (0 型) 语言类的真子集, 上下文无关 (2 型) 语言类是 CSL 类的真子集。

上下文有关文法 $G = (V, \Sigma, S, P)$, V 为变元的有限集, Σ 为终结符的有限集, $V \cap \Sigma = \varnothing$, S 为开始符号, $S \in V$, P 为产生式的有限集, P 中产生式呈 $\alpha \rightarrow \beta$ 形且 $|\alpha| \leq |\beta|$, $|\alpha| \neq 0$ (即 α 不是空字), 其中 $\alpha, \beta \in (V \cup \Sigma)^*$, $|\alpha|$ 表示字 α 所含符号的个数, $G = (V, \Sigma, P, S)$ 称为上下文有关文法 (CSG), G 生成的语言 $L(G)$ 必为某 LBA 所接受 (识别)。例如 $G = (\{S, A, B, C, D, E, H, K\}, \Sigma = \{a, \text{eps}, +, -, *, /, (,)\}, S$ 为开始符号, $P = \{S \rightarrow A, A \rightarrow B\} \mid a, B \rightarrow (C \mid B C \mid (A, C) \rightarrow D), CH \rightarrow DH, D \rightarrow HA \mid EA, E \rightarrow DK, H \rightarrow + \mid - , k \rightarrow * \mid / \}$, G 是一个 CSG, $L(G)$ 生成的是操作数为 a , 以中缀算子优先的算术表达式。CSG 的产生式可转化为 Kurada 形: $A \rightarrow a \mid B \mid BC, AB \rightarrow CD$, 其中 $A, B, C, D \in V, a \in \Sigma$; 或转化为右 (左) CSG, 其产生式为 $A \rightarrow a \mid B \mid BC, AB \rightarrow AC (AB \rightarrow CB)$ 形。这些文法是彼此等价的。右 (左) CSG 可用作 CSL 的字处理工具。特别当 $\Sigma = \{0, 1\}$ 时, CSG 为“标准化”的, 标准 CSG 的集合是可数的。CSL 集对并、交、连接、正闭包、替换、逆同态、CYCLE、Reversal 等运算封闭, 但对补、MIN、MAX 还不知道是否封闭 (封闭是指语言运算应用于 CSL 后结果仍为 CSL)。给完字 x , CSL L_1, L_2 , 正规语言 R (参见正规文法), $x \in L_1$ 是可以判定的, 但 $L_1 = \varnothing? L = \Sigma^? L_1 = L_2? L_1 \subseteq L_2? L_1 \cap L_2 = \varnothing? L_1 = R?$ 等都是不可判定的。

LBA 与 DLBA 接受的语言集是否相同是目前尚未解决的著名问题, 称为 LBA 问题。

参考文献

Hopcroft J E and Ullman J D. Introduction to Automata Theory, Languages, and Computation. Reading, Mass.: Addison - Wesley, 1979 (张一立)

xiangbian guangpan qudongqi

相变光盘驱动器 (phase change disc drive)

在相变光盘上进行数字信息记录与读出的设备。相变光盘驱动器具有写入、读出、伺服等功能, 主要由读写光学头、主轴、寻道跟踪和聚焦跟踪机构和电路, 以及读写通道及接口电路等部分组成。相变光盘的记录介质是相变材料, 它的记录原理是通过一种可逆变结晶材料在结晶与非结晶状态下不同的光特性来记录数据, 而这两种状态可以在高、中功率激光的照射下相互转换, 所以可以删除并重新写入数据。

相变光盘驱动器的记录和读出完全采用光学方法, 在写操作时, 聚焦激光束加热记录介质改变相变记录介质晶体状态, 用结晶状态和非结晶状态来区分 0 和 1; 读操作时, 利用结晶状态和非结晶状态具有不同反射率这个特性来检测 0 和 1 信号。

在进行记录时, 首先按数据中的 0 和 1 信号的转换来调制刻录激光的照射开关, 此时的激光是短时、高功率的, 被照射的记录介质的温度会迅速超过熔点, 在迅速冷却后就变为非结晶状态。此时, 结晶的地方可让光线通过到达反射层, 没有结晶的地方则很难让光线通过, 从而造成了激光反射功率上的差异。在进行数据抹除时, 用中等功率的激光对非结晶状态的记录介质进行相对长时间的照射, 使之慢慢达到结晶的温度, 从而转换回原来的结晶状态。

相变光盘驱动器的机械和伺服机构与磁光盘驱动器类似。相变光盘驱动器的数据传输速率为 1 Mb/s 左右, 平均寻道时间为 78 ms, 相变光盘采用盘盒结构, 有 80 mm 和 120 mm 两种, 主流相变光盘的容量为 650 MB, 已有 1.5 GB 的产品。

由于相变光盘的读出方法与 CD-ROM, CD-R 光盘相同, 因此兼容 CD-ROM 和 CD-R 的多功能相变光盘驱动器就容易实现。1994 年, 松下公司首先将相变型可擦写光盘驱动器与四倍速 CD-ROM 相结合, 推出了 650 MB 的相变光盘驱动器, 在一台光盘驱动器上同时实现了相变型可擦写存储与 CD-ROM 光盘读取功能。

相变光盘和 CD-RW 均使用相变记录技术, 但是由于松下公司的相变光盘采用了不同的数据格式, 因此这种光盘驱动器虽然可以读取 CD-ROM 光盘, 但是它却不能刻录 CD-R 光盘, 而且 CD-ROM 驱动器也无法读取松下格式的相变光盘。

与磁光盘 (MO) 技术相比, 由于相变光盘仅用光学技术来读写, 所以读写光学头可以做得相对比较简单, 存取速度也就可以提高。相变光盘驱动器

具有单光束直接重写能力,但所需的激光器功率比磁光盘的高,从而限制了相变光盘驱动器的转速比磁光盘的慢。

另外,相变光盘存储技术具有比磁光盘存储密度高、记录成本低、驱动器结构简单、读出信号信噪比高和不受外界磁场环境影响等突出优点,因此相变光盘技术已在 CD-RW, DVD-RW 和 DVD-RAM 驱动器上获得了非常成功的应用。

参考文献

干福熹等. 数字光盘存储技术. 北京: 科学技术出版社, 1998

(谢长生 黄浩)

xiangliang jisuan

向量计算 (vector computing) 以向量为单位进行的计算。相对于标量而言,向量是指一组同类型标量的有序集合,其中每个标量称为该向量的一个分量或元素。向量计算与标量计算的主要区别在于,一条标量指令仅处理一个或一对操作数,一条向量指令则可以处理 $n(n \geq z)$ 个或 n 对操作数,因此向量计算提供了更大的并行性。

向量计算机通常由一个标量处理机附加一个向量处理机组成。标量处理机负责所有指令的加载、译码和标量指令的执行,向量处理机负责向量指令的执行。按操作数的来源,向量计算机分为存储器-存储器型和寄存器-寄存器型。向量计算机对于有规则的大规模数值计算,大大提高了运算速度,主要依赖于下列技术的实现:采用流水线结构,促成运算的连续性;设置多个功能部件,并行处理多条向量指令;运用“链接”功能,使一个流水线部件的结果直接作为另一个流水线部件的操作数,不存在中间结果的存取问题;合理的存储器层次结构或通过流水存取方式提高存储器带宽,与运算部件的速度相匹配;减少程序中辅助指令条数,提高程序整体运行效率。

第一个向量计算机 CDCStar-100 出现于 1973 年。1976 年问世的 Cray-I 向量计算机被公认为世界上第一个超级计算机,每秒可产生一亿个浮点结果。此后,向量计算机成为高性能计算的主流产品,一直维持到 20 世纪 90 年代初期。其间,Cray 公司先后研制出 CrayXMP, CrayYMP, CrayC90 和 CrayT94 等。向量处理在其发展过程中,逐步与大规模并行处理相结合,出现了并行向量处理(PVP)体系结构,进一步提高向量计算机整体性能。PVP 体系结构通常以高性能互连网络连接多个结点,每个结点可包

含一个或多个高性能向量 CPU。NEC 公司的 SX 系列、富士通公司的 VPP 系列、Cray 公司的 SV1 和 SV2 都采用 PVP 结构。SX-5 和 VPP5000 均可达 512 个 CPU,峰值速度分别为 4 T FLOPS 和 4.9 T FLOPS。SV1 扩展型最多可达 1 024 个 CPU,峰值速度为 2 T FLOPS。2002 年 3 月日本 NEC 等单位宣布研制完成的“地球模拟器”是 21 世纪初期最快的并行向量超级计算机。该系统自行开发的向量 CPU 达到 8 G FLOPS,每个结点 8 个 CPU,全系统 640 个结点共 5 120 个 CPU,峰值速度 40 T FLOPS,存储器总容量 10 TB,互连网络为单层交叉开关。

另一个发展是标量处理器中加入了伪向量处理技术。所谓伪向量处理,是在足够存储器带宽的前提下,将数据从主存储器直接加载到浮点寄存器,避免因高速缓存容量问题引起供数不足的矛盾,借助滑动窗口技术实现每个时钟周期产生一个浮点结果的目的。日立公司的 SR2201 和 SR8000 系统采用了这项技术。

总体来看,向量计算机因其定制 CPU 和定制互连网络等因素,性能价格比远不敌商品化的高性能通用微处理器和互连产品构成的集群计算系统,自 20 世纪 90 年代以来一直在萎缩,这种趋势还将继续下去。唯有日本仍对向量机的发展给予相当重视,这是由于日本在传统上研究向量机的力量较强,以及日本研制向量处理器的微电子技术力量较强。

参考文献

1. 郑纬民, 汤志忠. 计算机系统结构(第二版). 北京: 清华大学出版社, 1998

2. Kai Hwang. Advanced Computer Architecture. McGraw-Hill-Inc., 1993

(桂亚东 谢向辉)

xiangmu guanli gongju

项目管理工具 (project management tool)

用来支持软件生产中项目管理活动的软件。通常的软件项目管理活动包括项目的计划、调度、通信、费用估算、资源分配以及质量控制等。软件生产是智力密集型的活动,其产品无物理外形,生产状态也“不可见”,因而难于检查和驾驭。软件项目管理工具就是要使这种生产过程成为可见、可控的过程。使用它能帮助进行成本估算、作业调度和任务分配,并制定出成本较低、风险较小的项目开发计划;同时能设法在预计工期和经费之内适当调整项目的安排,以节省时间和人力,从而对软件生产的各个环节进行严格、科学的管理,使项目开发活动获得最佳的

进程。

项目管理工具能对项目的任务调度、成本估算、资源分配、预算跟踪、人时统计、配置控制等活动给予帮助,它具有以下一些特征:(1)覆盖整个软件生存周期;(2)为项目调度提供多种有效手段;(3)利用估算模型对软件费用和工作量进行估算;(4)支持多个项目和子项目的管理;(5)确定关键路径,松弛时间,超前时间和滞后时间;(6)对项目组成员和项目任务之间的通信给予辅助;(7)自动进行资源平衡;(8)跟踪资源的使用;(9)生成固定格式的报表和剪裁项目报告。

项目管理工具通常都支持 PERT 和 Gantt 图。PERT 是计划评价与评审技术。该技术把网络方法用于工作计划安排的评审和检查。通常以带箭头的边表示活动,边的起讫结点表示活动的开始事件和结束事件,边的长度表示该活动的工作量或工期,各结点的顺序反映了各个活动在时序上的制约关系。利用 PERT 的网络图能求出关键路径和松弛时间,并能对计划的各个活动和资源分配等进行调整。Gantt 图是一种二维横道图,它广泛用于各种工程活动的进度计划管理。图的横坐标为时间轴,每个活动用一条水平线段表示,其起讫点对应的横坐标值即为该活动的开始和结束时间。

尽管新的项目管理方法和技术会改变人们已经习惯的工作方式,学习和掌握新工具也要花费一些时间,但是使用自动项目管理工具比用手工方法管理有许多优点,如:

(1)能对大型项目进行精确跟踪,使项目经理能及时掌握实际工作进展和实际资源消耗情况。

(2)能辅助开发 PERT, CPM(关键路径方法)和 WBS(工作分解结构),自动更新活动网络图和 Gantt 图。

(3)能自动计算、自动积累数据、自动生成图形和报表来取代人工计算、调度、统计和文档工作,提高管理工作效率。

国内外已有许多可用于项目管理活动的通用或专用的产品,如 Microsoft 公司的 Project, Lotus 公司的 Notes 和 Orgnizer, Primavera System 公司的 Primavera Project Planner 等。

参考文献

Thayer Richard (Ed). IEEE Tutorial on Software Engineering Project Management. Los Alamitos: Computer Society Press, 1988 (金茂忠)

xiaoxi chuli xitong

消息处理系统 (message handling system, MHS) 以存储转发方式传送具有各种内容和编码类型的消息的一种网络应用系统。它在电子邮件和电子数据交换 (EDI) 等方面获得了广泛的应用。

MHS 是国际电报电话咨询委员会 (CCITT) 在采用国际信息处理联合会 (IFIP) 的消息处理系统模型的基础上提出的,经过试行和修改,于 1984 年 10 月被标准化为 CCITT X.400 系列建议,并先后于 1988 年和 1990 年进行过修改和补充。该系列建议定义了 MHS 的模型、消息传输和交换的方式以及有关协议,它为分布式应用的通信提供了消息处理服务。表 1 给出了 X.400 系列的主要建议。

表 1 CCITT X.400 系列建议摘要

CCITT 建议	有关内容
X.400	消息处理系统及服务综述
X.401 (仅有 1984 年版本)	基本服务要素和任选用户设施
X.402	总体结构
X.403 (仅有 1988 年版本)	一致性测试
X.407	抽象服务定义约定
X.408	编码信息类型的转换规则
X.409 (仅有 1984 年版本)	传送语法和符号表示法
X.410 (仅有 1984 年版本)	远程操作和可靠传输服务
X.411	抽象服务定义和过程
X.413 (仅有 1988 年版本)	消息存储的抽象服务定义
X.419 (仅有 1988 年版本)	协议规范
X.420	人际消息交换系统
X.430 (仅有 1984 年版本)	Teletex (智能用户电报) 终端访问协议
X.435 (仅有 1990 年版本)	EDI (电子数据交换) 消息交换系统

MHS 的功能模型如图 1 所示,其最基本的构件是用户代理 (UA) 和消息传送代理 (MTA),另外,还包括访问单元 (AU) 和消息存储单元 (MS)。

对于电子邮件应用而言,UA 是一个用来实现与电子邮件系统接口的程序。它提供用户生成、发送和接收邮件的功能,并完成对邮箱的管理。

MTA 接收来自用户代理的邮件并保证将其发送出去,这就是电子化邮局。与常规邮政系统相似,消息在送到之前往往要经过多个邮局 (与之相应的是 MTA)。这样的多个 MTA 构成了报文传送系统 (MTS),以提供报文存储和转发服务。1 个 MTA 可接多个 UA, MS 或 AU。在某些场合,MS 也可与 MTA 合并在一起。

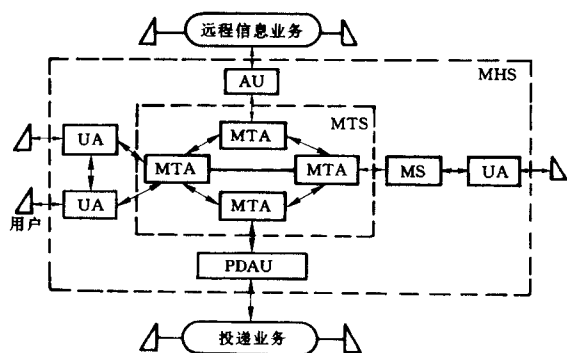


图1 MHS的功能模型

MS一般由投递消息和报告的存储消息数据库、登录输入数据库和登录输出数据库等3个数据库组成,故也可称为消息库。

MHS还可以通过相应的功能模块实现与公众电报网、用户电报网、智能用户电报网和邮政网相连接。

在实现MHS时,可根据具体情况对功能进行取舍。归纳起来,有3种不同类型的系统。第一种系统只有UA功能,第二种系统只有MTA功能,第三种系统既有UA,又有MTA功能。MHS中的UA通常由微型计算机系统实现;MTA既可以用微型计算机系统、工作站,也可以用大、中型计算机系统来实现。MTA通过局域网或分组交换网连到其他MTA。所用的协议结构是X.400系列协议结构。

X.400系列协议基本上是开放系统互连基准(参考)模型(OSI/RM)的应用层服务协议,因此它需要OSI/RM中1层~6层的实体提供完整的通信环境。MHS可分为两个子层:用户代理子层(UAL)和消息传送子层(MTL),其结构如图2所示。

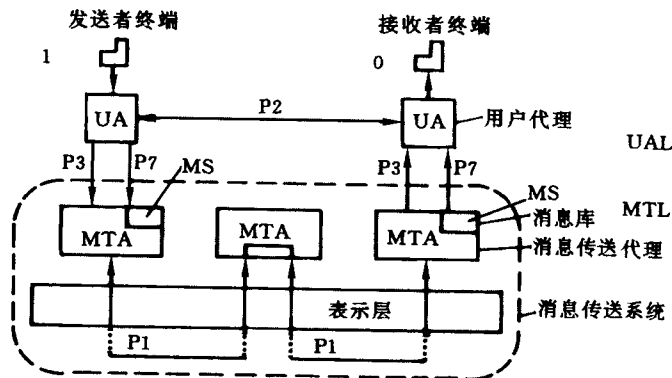


图2 MHS层次结构

用户代理子层UAL实现UA中有关通信部分的功能,由UAL形成便于传送的消息,以便在UA之间进行有意义的通信。该子层具有两种协议P2和P7。P2由X.420定义,它建立人际消息服务,主要有:①定义了一组协议要素,可用以构造用户代理协议数据单元(UA-PDU);②规定了与进行协议要素交换有关的操作;③指定了人际消息通信用户代理(IPM-UA)使用MTL服务所应遵循的准则。P7规定UA如何访问MS。

消息传送子层MTL向上层(即UAL)提供消息传送服务,保证在有限的时间内交付消息,需要时进行信息编码类型的转换。该子层也具有两种协议P1和P3。

P1由X.411定义,它规定了MTA间进行消息传送的协议规范,可实现3种服务:①可靠传送服务;②连接管理服务;③消息调度服务。

P3由X.410定义,它规定了UA和MTA或MS和MTA之间进行通信的协议规范,主要包括两部分内容:①连接管理;②操作,包括管理、提交、交付和通知操作。

参考文献

1. 曾华燊等译. 计算机网络. 成都: 成都科技大学出版社, 1989
2. 顾冠群. 计算机网络. 南京: 江苏科技出版社, 1989

(李学农)

xiaoxi chuandi

消息传递 (message passing) 在分布式并行计算机系统中,通过传递消息包来实现其中各计算机之间的通信和同步的一种机制。

一个消息可以是一个很短的同步信息,也可能是一个长度为数兆字节的数据文件。在通常情况下,消息是由一组消息包组成的。消息包是1次发送或接收的单位。它由包头、包体和包尾组成。包头由消息包的长度、目的结点编号、所传消息的编号以及消息内包的编号(包号)等信息组成。包体是真正要传送的数据,包尾则表示消息包结束的标志信息。

一个完整的消息传递过程包括发送、在互联网中包的传输和接收3个步骤。第二个步骤主要由硬件完成(参见路由选择),其余两个步骤由软、硬件共同完成。消息发送前需将消息分解并

按特定格式组成一个个消息包,放入发送结点缓冲区中,然后依次发送到互联网络中。目的结点从网络中取包、解包并组成消息。

消息传递中的两个最重要的问题是可靠性和效率。可靠性主要包含 3 层意思:消息包中个别位出错;消息包传丢或传到非目的结点以及消息传递过程中系统瘫痪。第一种情况可由专门的硬件或软件来发现和纠正。对于第二种情况,可以用适当的消息传递协议来保证将消息包正确地传送到目的结点。但是,这样做可能降低通信效率。第三种情况可能是死锁造成的,也可能是其他软件故障,情况比较复杂。这是大规模并行计算机系统必须解决的一个关键问题。

衡量消息传递效率的一个重要参数是消息包的延迟时间。它包括发送、网络传输和接收 3 部分时间。对于一个相当规模的互联网(例如 10×10 的网格),网络传输延迟一般不超过 $1 \mu\text{s}$,而一个发送或接收过程的操作系统额外开销可能超过数十微秒。降低这种额外开销的一个有效办法是使消息包中包含目的地址信息,并在消息传递中尽量减少操作系统的介入。

参考文献

Seitz C L. Concurrent Architecture, in Suaya and Birtwistle (eds), VLSI and Parallel Computation. San Mateo: Morgan Kaufmann. 1990, Chapter 3

(祝明发)

xiaoyin jishu

消隐技术 (hidden line/surface removal techniques) 对于三维空间的非透明物体,将观察者所看不见的棱线或表面消除的技术。

为了使计算机生成的图形具有在三维空间的真实感觉,将其看不见的物体部分即隐藏部分消除,这是真实感图形的最基本需求。如果不将隐藏的线或面消除,有时会产生对图形的错误理解。如图 1(a)是未经消除隐藏线的方块物体,它可以理解成图 1(b)所示的方块或图 1(c)所示的方块物体。

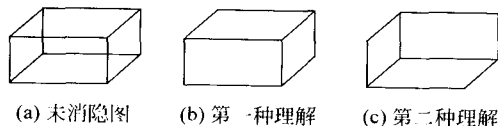


图 1 未消隐图形的歧义理解

当空间物体用线框方式(参见几何造型)绘制图形时,多面体用棱表示,将观察者所看不见的棱线消除,这时所涉及的消隐问题便是线消隐;当物体用具有不同明暗程度的面时,将观察者所看不见的面消除,所涉及的消隐问题为面消隐。消隐技术从 20 世纪 60 年代即开始进行研究,当时由于图形显示器是线产生器装置,因而消隐问题涉及的主要是线消隐;自从 60 年代后期光栅图形显示器得到广泛应用后,面消隐问题便相应地受到更多关注。线及面消隐技术是计算机真实感图形生成的最基本、最重要的技术之一,20 多年来研究并实现了许多成功的消隐算法。尽管如此,消隐技术仍然是今天人们所关注的问题。在计算机图形学发展的若干新兴领域,如多媒体计算技术、虚拟现实等,当实时动态的计算机真实感图形生成成为必要时,快速有效的消隐成为十分重要的问题。这时,如何利用计算机软、硬件实现快速的实时消隐成为这些领域的瓶颈技术之一。

线消隐的主要技术有 Roberts 算法等。理论上说,对于物体的每条棱边线段,只要求出它与每个多边形的遮挡关系并进而求出相应的隐藏线段,线消隐问题便可以迎刃而解。然而这种通用的解决办法实际上是难于在实际中应用的,因为它所需要的计算量太大。Roberts 算法针对这一问题采取了有效的解决办法以提高效率。它的核心思想是,根据组成物体多面体的面与面以及面与线之间的相对关系逐步分批地求出隐藏线或隐藏线段。它将所有物体按视线方向从远至近进行排序;从最远的物体开始,对于每个物体,检查和确定与其他每个物体的遮挡关系并计算隐藏线。

面消隐的主要技术有缓冲器算法、扫描线算法以及表优先级算法等。Z-缓冲器算法是利用记录图形深度的缓冲存储器实现面消隐的一种简单实用算法。它在图像空间实施消隐。该算法利用一个记录投影方向上每个像素上物体投影深度值的缓冲存储器(Z-buffer),每当物体投影时,每个像素上的投影深度将与此像素在 Z-缓冲器中已记录的深度进行比较。只有此深度比所记录的深度小时,才将相应点上此物体的颜色或灰度值保存或显示出来。扫描线算法是使用逐行的像素扫描线实行图形绘制和面消隐的一种算法。该算法在图像平面逐行扫描,在每条扫描线上确定和计算可见的扫描线段,即进行消隐处理。扫描线算法还可以与 Z-缓冲器算法结合起来。即在每条扫描线上实现 Z-缓冲器算法。表优先级算法是按照物体离视点的远近进行优先级

排序从而对图形实施有序绘制的一种消隐算法。该算法的处理过程与画家创作一幅油画类似:先画远景,再画中景,最后画近景。因而该算法在处理简单元素(如多边形)时常被称为油画算法。

参考文献

1. 唐泽圣,周嘉玉,李新友. 计算机图形学基础. 北京:清华大学出版社,1995
2. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2001

(吴恩华)

xiaonao wangluo moxing

小脑网络模型 (cerebellar model articulation controller, CMAC) 根据小脑的生物模型提出的一种神经网络模型。又称 CMAC 网。

CMAC 模型的基本思想可表述如下。设输入空间为 X , x 为属于 X 的任一输入向量。我们将输入空间划分为 N 个相互重叠的小格。对应于第 i 个小格定义函数 $\varphi_i, i=1, 2, \dots, N$, 其中

$$\varphi_i(x) = \begin{cases} 1 & \text{若 } x \text{ 属于第 } i \text{ 个小格} \\ 0 & \text{其他} \end{cases}$$

如果我们给每个小格 i 赋予一个权值 w_i , 那么对于任一输入向量 x , 相应的输出 y 可写成

$$y(x) = \sum_{i=1}^N w_i \varphi_i(x)$$

从神经元角度看,每个小格相当于一个神经元。每个神经元的感受域即为小格所占据的那部分输入空间。如果输入向量 x 落到某个神经元的感受域中,那么这个神经元就被触发。由于小格相互重叠,因此神经元的感受域也相互重叠。于是一个输入向量可能会同时触发多个神经元。所有被触发的神经元对应的权值加在一起即为这个输入向量所对应的输出。

根据小脑网络模型的上述思想,可以给出该模型的结构,如图 1 所示。图中 S 表示输入空间, A 表示存储体。 A 的每个存储单元保存一个权值。CMAC 映射算法将 S 中的每个输入向量映射到 A 中 g 个存储单元。这 g 个存储单元的内容之和即为 CMAC 的输出。若两个输入向量在 S 中距离较近,则它们映射到 A 的相应存储单元有重叠,距离越近,重叠单元越多。若两输入向量在 S 中距离足够远,则它们映射到 A 的相应存储单元无重叠。这就是 CMAC 的局域泛化特性, g 称为泛化常数。

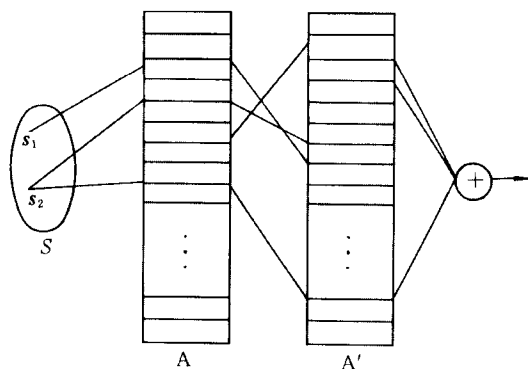


图 1 小脑网络模型

根据 CMAC 映射算法,每个输入向量 s 在存储 A 中激活 g 个地址。我们用向量 X 表示 s 所激活的地址, X 的每个元素定义如下

$$X(i) = \begin{cases} 1 & \text{若 } A \text{ 中第 } i \text{ 个地址被 } s \text{ 激活} \\ 0 & \text{其他} \end{cases}$$

我们称 X 为指示器。所以常常使用 X 代替 s 做为 CMAC 的输入。

令 W 表示 A 的权值向量,则 CMAC 的输出可表示为

$$y = X^T W$$

CMAC 的学习算法可表示为

$$W_{k+1} = W_k + \alpha_k X_k^T (y_k - X_k^T W_k) / g$$

式中 X_k 是第 k 次学习的输入;

y_k 是期望输出;

W_{k+1} 是第 k 次学习之后的权值向量;

α_k 是学习速度;

g 是泛化常数。

注意到 $|X_k| = g$, 令 $\beta_k = g\alpha_k$, 则上式可写成

$$W_{k+1} = W_k + \beta_k (y_k - X_k^T W_k) X_k / |X_k|^2$$

很容易看出,CMAC 算法也是一种最小均方算法。它的模型可分为两部分。前端是一个固定的非线性映射,将任一输入向量 s 映射到 A 的地址向量 X 。后端是一个自适应线性元件,它的权值通过学习可以改变。在实际问题中,输入空间往往很大,相应要求存储 A 也很大,应用中很难实现。但幸运的是,大多数学习问题不涉及空间的全部。因此在 CMAC 模型中,往往利用杂凑编码将存储 A 映射到一个较小的易于实现的存储 A' 。存储 A 称为虚拟存储, A' 称为杂凑存储或物理存储。

CMAC 模型是 Albus 于 1975 年首先提出的,它具有学习速度快,无局部极小点和局域泛化等良好

性质。近年来逐渐引起人们的兴趣并在许多领域得到了广泛应用。比如,使用一种改进的 CMAC 模型对混沌系统的时间序列进行预测(1989 年)。在 CMAC 模型中引入 Kohonen 的自组织算法;根据输入样本分布对 CMAC 的存储机制做自适应的修正(1990 年);研究了 CMAC 模型的容错特性;将基于 CMAC 的控制器与传统的自适应控制做了比较(1990 年);将 CMAC 用于机器人的实时控制(1987 年,1989 年,1990 年),取得令人鼓舞的结果。在理论方面,Cotter 等讨论了 CMAC 模型的计算能力,给出了两类 CMAC 模型不能实现的函数(1992 年);建立了 CMAC 学习算法与线性方程组的 Gauss-Seidel 迭代算法的等价关系(1992 年),并由此证明了在循环学习方式下学习速度为 1 时 CMAC 学习算法的收敛性。

CMAC 网无论在理论上还是实用上都有许多待进一步研究的课题,尤其在自动控制领域的应用有很大潜力。

参考文献

Albus J S. A New Approach to Manipulator Control: the Cerebellar Model Articulation Controller (CMAC). Journal of Dynamic Systems, Measurement and Control, Sept., 1975:220 ~ 227 (张钹 姚殊)

xiaoxing jisuanji

小型计算机 (minicomputer) 规模介于微型计算机和大型计算机之间的一种计算机。

发展简史

小型计算机的发展大致可以划分为 4 个阶段:第一阶段为 20 世纪 60 年代中后期。这个阶段的特点是采用晶体管或小规模(少数采用中规模)集成电路的非系列化计算机,字长不统一,软件不完善。一般用于实时过程控制、数据采集和数据处理。典型的产品如 1965 年 DEC 公司推出的 PDP-8,采用中规模集成电路,字长为 12 位,IBM 公司的用于科学计算的 IBM-1130 和用于过程控制的 IBM-1800。第二阶段为 60 年代末至 70 年代末。这个阶段的特点是采用中、大规模的集成电路的系列化计算机,软件的功能、性能和品种都有较大的发展,系列化保证了更新的机型可以使用老机型支持的软件和外部设备等,即具有兼容性。典型的产品有 DEC 公司的 PDP-11 系列,其中的 PDP-11/70 采用高速缓存,在保留单总线结构的同时,与外围设备之间有 32 位专用数据通路,以确保输入输出的高速度。另外一个

较有影响的小型计算机系列是 Data General 公司的 NOVA 系列。第三阶段为 70 年代末至 80 年代末。这个阶段的特点是采用大规模、超大规模集成电路;形成了结构开放的硬件和软件标准;开发工具和环境的形成以及 UNIX 作为小型计算机操作系统的地位得到确立。典型的产品如 VAX 11 系列, MICRO VAX 系列和王安 VS 系列等。第四阶段为 80 年代末以后。主要特征是处理机体系结构出现了重大变革,精简指令集计算机大量出现和软件工业标准的建立,导致小型计算机性能的大幅度提高和应用的快速发展。

分类和应用

小型计算机的分类方法很多,以其字长可分为小型计算机(8 位,16 位)和超级小型计算机(32 位,64 位);按处理器的体系结构可分为精简指令集计算机和复杂指令集计算机;按对出错的反应可分为容错计算机和非容错计算机;按其用途可分为通用计算机和专用计算机,也可分为民用计算机、工业用计算机和军用计算机。工业用计算机和军用计算机一般使用环境比较恶劣。为适应振动、冲击、辐射、温度、湿度以及灰尘、盐雾、微生物等环境要求,可分为加固型计算机和非加固型计算机。

小型计算机的应用领域十分宽广,例如,数据采集和数据处理,工程设计和科学计算,信号处理和图像处理,工业过程控制,武器控制、模拟和训练系统,企业管理以及在客户-服务器结构中用作服务器等。

参考文献

克拉夫特 G D,托伊 W N 著. 小型/微型计算机硬件设计. 陈森锦,王希仁译. 北京:科学出版社,1984 (方金仰)

xie chuliqi

协处理器 (coprocessor) 在计算机中,协同主处理器执行某些特定操作的专用处理器。

协处理器使计算机在执行这些特定操作的时候,可以获得更好的性能。协处理器是可选件,集成在一个单独的芯片上。典型的例子有:数值运算处理器用来执行浮点数运算或其他运算。输入输出处理器用来传送输入输出数据。协处理器扩充了可供程序员使用的指令系统,当主处理器接收了一个它不直接支持的操作时,便将控制转移到相应的可以支持该操作的协处理器上。协处理器往往能以很快的速度完成交付给它的任务,使得主处理器可以腾出时间去执行必须由它完成的任务。在一个计算机

系统中,可以使用多个协处理器。例如,一个协处理器实现高速数值计算,另一个协处理器实现数据库管理的原语。

协处理器的概念是从**附属处理机**发展而来的。附属处理机使计算机系统对不同类型的用户都具有较好的性能价格比。一个计算机可划分成不同的模块,其中,将能满足大多数用户的常规需求的模块作为主处理机,针对一些特殊用户的需求而设计的专用模块称为附属处理机。这些专用模块在执行特殊用户的特定算法时,可以获得高效率。对于某些特殊的用户,用通用性强的主处理机和特定算法能获得高效率的附属处理机,通过高速通路连接起来,构成适合于这些特殊用户的专用计算机系统。它既具有通用性强的主处理机由于规模生产所带来的低成本的优点,又可用增加不多的代价获得附属处理机在执行该特定算法时带来的高效率。最常见的附属处理机是向量处理机。美国的 FPS 公司曾经是专门生产附属向量处理机的公司,它所生产的 AP-120B 和 FPS-164 等附属向量处理机,作为向量部件连接到通用主机(例如 DEC 公司的超级小型计算机)上,使那些需要进行大量向量计算或矩阵运算的用户获得性能价格比很高的计算机系统。此外,一些著名的计算机公司如 IBM、富士通、日立和 NEC 等,也生产附属向量处理机(或称向量部件或阵列部件)与自己的通用计算机配套。

随着微电子技术和计算机技术的飞速发展,主处理机模块和附属处理机模块可以分别做在超大规模集成电路芯片上,称为**微处理器**和**协处理器**。跟附属处理机与主处理机系统相比,协处理器与主处理器之间的耦合程度要紧密得多,从而使主处理器管理协处理器所需的系统开销小得多。

协处理器的典型例子是 Intel 公司的 x 87 系列数值运算协处理器。以 Intel 80387 数值运算协处理器(以下简称 387)为例,它是为 Intel 80386 微处理器(以下简称 386)专门设计的,用来高效执行高精度的浮点运算,以及正弦、余弦、正切、对数等超越函数的计算。387 协处理器与 386 微处理器并行连接可以构成一个高性能的 386 微处理机系统。387 协处理器为 386 微处理器扩充了七十多条指令(称为 ESC 指令,指令的最高 5 位为 11011),还扩充了多种数据类型,如:32 位、64 位和 80 位浮点数;32 位和 64 位整数,以及 18 位二进制编码的十进制数,浮点数遵循 IEEE 754 浮点数标准。387 在硬件方面为 386 扩充了 8 个 80 位的浮点寄存器,以及 16 位的控

制寄存器、状态寄存器和标志寄存器。系统启动后,387 需完成初始化过程,它与 386 执行同一个指令流,共同完成对指令的译码。当 386 译码出一条 ESC 指令后,386 将取得的 387 操作码或 387 的操作数通过专门为 387 设定的 I/O 端口(800 000 F8 H ~ 800 000 FFH)传送给 387。接着 386 与 387 即可并行工作,386 通过 BUSY 引脚实现与 387 的同步,通过 ERROR 引脚检测 387 需进行的异常处理。这种协同工作方式,大大减少了软件系统的开销。此外,387 的所有存储器的访问均由 386 负责管理。所以,不论 386 是工作在实地址、保护态或者虚拟 8086 状态,387 都以同样方式工作。387 只负责对 386 传送给它的指令和数据进行操作,不必顾及 386 的工作模式,从而简化了 387 的设计。

协处理器的出现是微电子技术和计算机技术发展一定阶段的产物。随着微电子技术的进一步发展,原来用单独的芯片实现数值协处理器的功能,已经可以与主处理器在同一个芯片中实现。如 Intel 80486DX 和 Pentium 已包含数值协处理器,于是单独的数值计算协处理器已逐渐消失。

参考文献

1. Protopapas D A. Microcomputer Hardware Design. Englewood Cliffs, New Jersey: Prentice Hall Inc., 1988
2. Valerie Illingworth etc. Dictionary of Computing. 2nd Edition. New York: Oxford University Press, 1986
3. Intel Corp., Microprocessors Vol. 1. 3065 Bowers Avenue, Santa Clara, CA95051, 1995

(韩承德)

xietong licheng

协同例程 (coroutine) 可以和其他例程互相调用的例程,它们彼此处于平等地位,调用后无须返回到开始位置,且自带工作区。

协同例程由两部分组成:不受每次执行影响的固定部分(如机器语言指令)和随不同执行而变的可变部分(如数据和控制信息)。可变部分又称工作区。

协同例程与子例程的区别在于工作区的生存期不依赖于控制进入或离开例程的时间。而且协同例程的工作区需保持局部指令计数器的内容,以便每当再次进入一个例程时,总是从它最近离开的那次执行被停止处开始继续执行。图 1 中的 C1 和 C2 表

示两个协同例程。

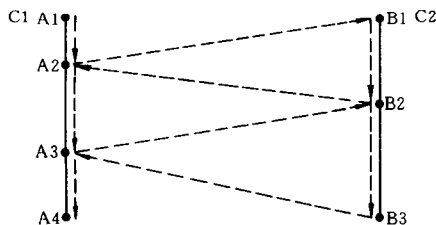


图 1

在一台处理机上,例程 C1 从起点 A1 开始,当执行到 A2 处被中断,欲调用例程 C2,就必须先将例程 C2 的执行点 A2 的现场保存起来,然后才进入例程 C2 的起点 B1。当 C2 执行到 B2 位置又调用 C1 时,则借助先前保存的现场如 A2 的位置,继续在曾被中断的执行点 A2 执行下去,到了 A3 位置又调用 C2。以此类推,直到 C1 执行到终点 A4 处为止。

下面介绍一个著名的“快速分类”算法。其思想是:对一组数组元素 $A[m], \dots, A[n]$, 使这些元素分成两部分:对某个合适的 j , 左部 $A[m], \dots, A[j-1]$ 包含小于某个值 V 的所有元素, 右部 $A[j+1], \dots, A[n]$ 包含大于 V 的所有那些元素, 位于这两部分之间的元素 $A[j]$ 将等于 V 。划分进行如下:依次从左到右扫视各个元素,直到找到一个大于 V 的元素为止,这时便从右到左扫视,直到找到一个小于 V 的元素为止。然后,再从左到右扫视,如此等等,每次把错位元素放到相反的一边,直到两种扫视相会时为止。这种算法可以写成一个例程(如,引进布尔变量以区分从左扫视与从右扫视),也可写成两个协同例程。可以说从概念上来讲后者较为简单明了,具体程序如下(先赋初值,然后调用 `move i`):

```

coroutine move i; 从左到右扫视
  loop: if A[i] > V
  then A[j]: = A[i];
  resume move j;
  fi;
  i: = i + 1;
  while i < j repeat;
coroutine move j; 从右到左扫视
  loop: if V > A[j]
  then A[i]: = A[j];
  resume move i;

```

```

fi;
j: = j - 1;
while i < j repeat;
i: = m; j: = n; V: = A[j];
call move i;
A[j]: = V; 当 i = j 时

```

其中:语句 `resume move j` 在将控制转移到协同例程 `move j` 之前得先保存 `move i` 当前执行点的现场。同时, `resume move i` 在将控制转移到协同例程 `move i` 之前也得先保存 `move j` 当前执行点的现场。

参考文献

Hansen P B. Operating System Principles. Prentice Hall, 1973 (段祥)

xinyidai wangji xieyi

新一代网际协议 (new generation internet protocol, IPv6) 为适应 Internet 飞速发展,满足其扩展地址空间等各种应用需求而设计的新的网际协议。

网际协议第 4 版 (IPv4) 为 TCP/IP 协议集和整个 Internet 提供了基本的通信机制。自 1970 年发布以来,已沿用至今。虽然 IPv4 的设计是比较完善的,但随着技术和应用的发展,已不能满足需要。尤其是 Internet 用户的飞速增加,将导致 IP 地址(参见 Internet 地址)空间很快耗尽,因此迫切需要对 IPv4 进行更新。在最初设计 IP 时,32 位地址空间是很充裕的,但如今,32 位 IP 地址空间已不能满足 Internet 的飞速增长。

除了地址空间需要扩展以外,还有一些其他因素也要求改变现有的 IP 设计,主要是日益增加的各种新的应用需求。例如,实时语音(言语)和图像通信要求新版的 IP 能为特定应用预留资源以降低延迟。又如一些新的应用需要安全通信,因此要求新版的 IP 应具有鉴别发送者的安全机制。

新版的 IP 已正式命名为 IPv6,它保持了 IPv4 许多成功的特点:支持无连接传送;允许发送方选择数据报大小;要求发送方指明数据报在到达目的站前的最大跳步数。IPv6 保留了 IPv4 中的大多数选项,其中包括分段和源站路由选择。

IPv6 对 IPv4 也做了许多修改。IPv6 的修改可分成五大类:

(1) 更大的地址空间 这是 IPv6 最显著的变化。IPv6 将原来 IPv4 的 32 位地址空间增大到 128 位地址空间。IPv6 的地址空间是足够大的,在可预

见的将来是不会耗尽的。

(2) 灵活的报头格式 IPv6 使用一种全新的、不兼容的数据报格式。在 IPv4 中,使用固定格式的数据报报头,在报头中,除选项以外,所有的字段都在一个固定的偏移位置上占用固定数量的八位组,而 IPv6 使用了一组可选的报头。

(3) 增强的选项 IPv6 允许数据报包含可选的控制信息,包含了 IPv4 不具备的选项,提供了新的功能。

(4) 支持资源分配 IPv6 提供一种新的机制,允许对网络资源预分配,取代了 IPv4 的服务类型说明。这些新的机制支持实时语音(言语)和视像等应用,保证一定的带宽和延迟。

(5) 支持协议扩展 IPv6 的一个很重要的改变是该协议允许新增特性,而且不需描述所有细节。这种扩展能力使协议能适应底层网络硬件的改变和各种新的应用需求。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999

2. Douglas E Comer. Internetworking with TCP/IP, Volume I, 3rd ed. Prentice Hall Inc., 1995

(胡道元)

xinnian

信念(belief) 未经证实但有一定合理性的假设性知识。说“未经证实”,是指它成立的证据尚不充分;说“有一定合理性”,是指它确实一定程度上得到了某些证据的支持。例如,“外星球上存在高级动物”就是一种信念。

演绎推理固然严格,但它的源头——公理,却只能是信念。演绎推理无法从本质上扩展知识。我们的知识之所以在不断扩展,是因为我们在证据的支持下不断产生新的信念。

使计算机具有非演绎推理的能力,实际上就是赋予计算机信念和处理信念的能力。这种能力在知识处理系统和决策支持系统所起的作用,无论怎样强调也不会过分。

在人工智能介入信念的研究之前,逻辑学界已对信念的静态性质作了很多研究,其中以 Carnap 为代表的概率逻辑学派着重研究证据对信念的支持关系;以 Rescher 为代表的模态逻辑学派则着重研究一个认知主体的信念之间的关系。人工智能关于信念的研究偏重于一个信念体系在与证据流

的相互作用中动态演化的内在规律。比较著名的工作有 Shafer 的证据理论和 Pearl 的 Bayes 信念网络理论等。

从人工智能角度看,信念的处理主要集中于下面三个环节:

(1) 信念的生成 比较成熟的信念生成机制包括:

归纳 这是一种基于普遍性,由个别推及一般的信念生成机制。

类比 这是一种基于相似性,由个别推及个别的信念生成机制。

反绎 这是一种基于因果性,由结果推及原因的信念生成机制。

此外还有联想。这是一种基于概念连接和替换的、由此推彼的信念生成机制。

(2) 信念的评价 扩大知识是要付出代价的,这代价就是信念的可错性。如何在事前根据已知证据估计一个信念的正确程度(或者说犯错误的风险),构成关于信念的评价的研究。目前评价信念函数的种种度量体系在人工智能中被称为不确定性度量,主要有概率,Shafer 证据理论中的信念函数,模糊理论中的隶属度函数,MYCIN 中的确信因子 CF 等。不同的度量体系体现了关于信念的不同的认知合理性标准,断言哪一种最合理,目前还为时尚早。

(3) 信念的修正 随着证据的不断积累,原有的一些信念可能被证明为错误的或不合理的,另一些新的、更为合理的信念则需要加入到信念体系中来,这就构成信念内容的动态调整及演化,也就是通常所说的“信念修正”。考虑到证据积累是一个渐变的过程,有时虽然信念内容不变,但对信念的评价却在不断地变化,这是另一种意义上的“信念修正”。属于前一类“修正”的工作有 Doyle 的 TMS 机制,Gärdenfors 的知识流理论等。属于后一类“修正”的工作有 Bayes 信念网络中后验概率的修正及 D-S 证据合成公式所描述的机制。

目前尚不存在关于信念的涉及上述三个主要环节的统一理论,学术界对于信念的认知合理性问题也还有不同的意见和标准。在实用中,知识处理系统和决策支持系统的设计者们一般都是根据自己的问题领域的具体特点来选择某一种适合自己的方案。尽管如此,由于信念及处理信念的能力在人工智能中的极其重要的意义,这方面的研究将继续深入下去,并可望获得良好的应用前景。

参考文献

1. Shafer G. A Mathematical Theory of Evidence. Princeton University Press, 1976
2. Rescher N. Epistemic Modality: The Problem of a Logical Theory of Belief Statements. In: Topics in Philosophical Logic, D. Reidel Publishing Company / Dordrecht - Holland, 1968 (白硕)

xinxi

信息 (information) 人们按照已知的约定对数据赋予的含义。信息是继物质、能量之后,人类发现的认识自然、改造自然的新资源。信息已渗透到人类社会的各个方面、各门学科。社会信息化已成为人类的追求目标,它是社会发展的必然。社会信息化的主要特征是计算机及计算机服务业面向人人。

以信息为研究对象的学科称为信息科学或信息学,其内容涉及信息获取、表示、储存、传输、处理、应用等方面(参见程序)。

参考文献

- IEEE standard glossary of software engineering terminology. New York, N. Y. USA. IEEE, 1990
(徐家福)

xinxi jiansuo fangfa

信息检索方法 (information retrieval method) 有关信息检索的数学模型和系统效用评价的方法。

信息检索系统(IRS)具有文献存储、标引、查询和管理等功能,它和关系数据库管理系统(RDBMS)在操作的数据对象和主要的操作类型上有很大的区别,信息检索(IR)处理的对象是非结构化的文献,而关系数据库管理系统(RDBMS)处理的对象是结构化数据——表;在数据操作方面,IR中的检索是非精确的和基于概率的,存在查全率和查准率的关键性问题(查全率是检索出的相关文献数与数据库中总的相关文献数之比;查准率是检索出的相关文献数与检索出的文献总量之比),而RDBMS中的查询是确定性的。

信息检索模型是信息检索理论研究的主要课题。迄今提出的信息检索模型主要有布尔检索模型、向量空间模型、概率检索模型、扩展布尔模型和语义模型等。

布尔检索模型提供“与”、“或”、“非”3种逻辑操作,针对全文检索的需求,又扩充了位置检索功能,如“同一段”、“同一句”等。由于布尔模型概念简单,实现容易且检索效率高,因此为商用系统普遍采用。但布尔模型有重大缺陷:①命中量很难控制;②难以实现有效的相关排序;③布尔提问式有时会产生违反常规的结果。

向量空间模型用向量来表示词、文献和提问,通过计算文献所对应的向量与提问所表示的向量的相关程度来获得检索结果。Cornell大学研制的SMART系统是向量检索模型的代表。向量检索模型的缺点是计算复杂度很高。

概率检索模型的基本前提是针对某一给定的提问,在先前检索出的相关文献中出现的词应该较那些未出现的词更为重要。目前已有基于概率模型的实用系统,但概率模型存在参数准确估计的困难。

扩展布尔模型融合了其他几种模型的优点,可以实现相关排序。

20世纪80年代后期,借助人工智能,特别是自然语言处理的研究成果,还提出了语义模型。

现代检索模型的共同特征是使用非精确检索技术和能对检索结果进行相关排序,从而实现按内容,而不是仅仅按关键词的出现与否来进行信息检索。

检索系统的评价也是信息检索理论的重要组成部分,查全率和查准率是评价检索系统的两个最重要指标,其他评价因素包括速度和空间占用等。研究表明,查全率和查准率呈互逆的增长关系。评价研究必须基于公认的测试文献集。

参考文献

1. Salton G, McGill M J. Introduction to Modern Information Retrieval. McGraw Hill, 1983
2. Salton G. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison Wesley, 1989
3. Frakes W B, Yates R B. Information Retrieval: Data Structures and Algorithms. Prentice Hall, 1992 (施水才)

xinxi jiansuo zhong de xiangguan fankui

信息检索中的相关反馈 (relevance feedback in information retrieval) 根据对检索结果的相关判断,自动修改检索方式进而获得更加满意的检索效果的过程和技术。

相关反馈的研究大都基于具有相关排序能力的检索模型(如向量空间模型、概率检索模型),但近来已有部分研究结果被修正以适应布尔检索模型(参见信息检索方法)。

相关反馈有两种基本方法:第一种是重新确定词权值,即在不修改原表达式的前提下,根据用户的相关判断及提问词在相关及非相关文献中的分布规律,重新确定检索词权值。将在相关文献中出现的词之权值增加而将未出现者减少,然后再次进行相关排序确定检索结果。事实上这是概率模型的基本形式。第二种是提问修正,修改提问中实际出现的词。这是因为用户所给出的查询表达式中经常含有其所需文献中不存在的词,并且总有一些未检索出的相关文献之标引词,与查询表达式及其他相关文献标引词不同。这样无论如何重新加权,对于检索那些以原始提问中未出现的词标引的相关文献均无帮助。因此,如果原来提问不太合适,必须修改初始提问来提高性能。这种方法的关键在于帮助用户决定如何修改提问。为了避免随意性,修改必须在系统的引导下进行,即系统给出适当的帮助信息,用户依信息作出判断并完成修改。

参考文献

1. Salton G, McGill M J. Introduction to Modern Information Retrieval. McGraw Hill, 1983
2. Salton G. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison Wesley, 1989
3. Frakes W B, Yates R B. Information Retrieval: Data Structures and Algorithms. Prentice Hall, 1992
(施水才)

xinxi jiaohuan yong hanzi bianma zifuji
GB 2312—1980

信息交换用汉字编码字符集 GB 2312—1980
(Chinese Ideograms Coded Characters Set for Information Interchange-Basic Set, GB 2312—1980) 1981年我国颁布的第一个汉字编码国家标准,全称为《信息交换用汉字编码字符集·基本集》(GB 2312—80)。

GB 2312 国标字符集由3部分组成。第一部分是字母、数字和各种符号,包括拉丁字母、俄文、日文平假名与片假名、希腊字母、汉语拼音等共682个(统称为GB 2312图形符号);第二部分为一级常用

汉字,共3755个,按汉语拼音排列;第三部分为二级常用字,共3008个,因不太常用,所以按偏旁部首排列。汉字总数为6763个。

GB 2312 国标字符集构成一个二维平面,它分成94行、94列,行号称为区号,列号称为位号。每一个汉字或符号在码表中都有各自的位置,字符的位置用它所在的区号(行号)及位号(列号)来表示(图1)。

区号	位号	①	②	③	...	⑨4
①		字母、数字和各种符号				
⑨		一级汉字 (3 755个)				
⑩6						
⋮						
⑤⑤		二级汉字 (3 008个)				
⑤6						
⋮						
⑧7		— — — — — (扩充使用)				
⋮						
⑨4						

图1 GB 2312—80 字符集的组成

在计算机内部,为了处理与存储的方便,每个汉字的区号和位号分别用单字节来表示,例如,“大”字的区号是20,位号是83,它的区位码是2083,用双字节表示为:00101000 01010011。

区位码不能用于汉字的通信。为了避免与ISO 2022中用于通信的控制码(00H~1FH)发生冲突,每个汉字的区号和位号必须分别加上32(即二进制0010 0000)。经过这样处理得到的代码称为汉字的“交换码”。因此,“大”字的交换码是:00110100 01110011。

由于文本中的汉字与西文字符经常混合在一起使用,汉字信息如不予以特别的标识,它与单字节的标准ASCII码就会混淆不清。为了解决这个问题,采用的方法之一就是把一个汉字看作两个扩展ASCII码,使表示GB 2312汉字的两个字节的最高位都等于“1”。这种高位为1的双字节(16位)汉字编码就称为GB 2312汉字的“机内码”,又称内码。目前,这种表示方式已经成为GB 2312汉字内码的一种事实上的标准。上面所说的“大”字的内码是:

10110100 11110011 (B4F3)

不难看出,GB 2312 汉字内码在双字节代码空

间中,其码位分布于右下角的 1/4 象限,如图 2 所示。

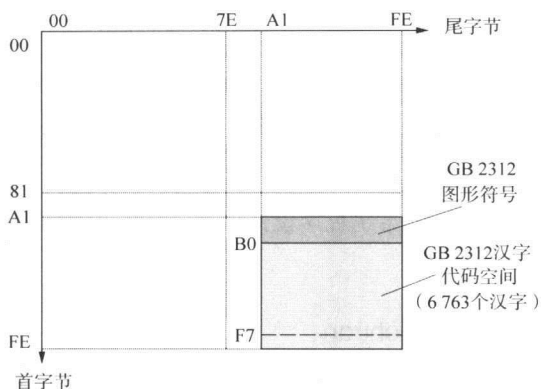


图2 GB 2312—80 汉字内码在双字节代码空间中的码位分布

参考文献

张福炎,孙志挥. 大学计算机信息技术教程. 南京:南京大学出版社,2003 (张福炎)

xinxi jiaohuan yong hanzi bianma zifuji de kuochong, GB 18030

信息交换用汉字编码字符集的扩充 GB 18030(Chinese Ideograms Coded Charac-

表1 GB 18030—2000 码位范围的分配

字节数	码位空间				码位数
单字节	0x00 ~ 0x80				129
双字节	第 1 字节		第 2 字节		23 940
	0x81 ~ 0xFE		0x40 ~ 0x7E, 0x80 ~ 0xFE		
四字节	第 1 字节	第 2 字节	第 3 字节	第 4 字节	1 587 600
	0x81 ~ 0xFE	0x30 ~ 0x39	0x81 ~ 0xFE	0x30 ~ 0x39	

GB 18030—2000 标准采用单字节、双字节和四字节 3 种方式对字符编码。单字节部分使用 0x00 至 0x80 共 129 个码位,与 7 位的标准 ASCII 字符的编码保持一致,并增加了一个欧元符号。

双字节部分收录内容主要包括 GB 13000—1 全部 CJK 汉字 20 902 个、有关标点符号、表意文字描述符 13 个、增补的汉字和部首与构件 80 个、双字节编码的欧元符号等。

四字节部分收录了上述双字节字符之外的,包括 CJK 统一汉字扩充 A 在内的 GB 13000—1 中的全部字符。

ters Set for Information Interchange - Extension for the Basic Set, GB 18030) 2000 年我国颁布的一个新的汉字编码国家标准,全称为《信息交换用汉字编码字符集的扩充集》(GB 18030—2000)。

UCS/Unicode 编码中的汉字及其编码与我国已使用多年的 GB 2312 和 GBK 标准并不兼容,为了既能尽快地向 UCS/Unicode 编码标准过渡,又能向下兼容 GB 2312 和 GBK 汉字编码标准,信息产业部和国家质量技术监督局在 2000 年联合发布了 GB 18030—2000 汉字编码国家标准,并在 2001 年开始执行。

GB 18030—2000 标准在字汇上与 UCS/Unicode 保持兼容,从而为中文信息在 Internet 上的传输与交换提供了保障。该标准同时收录了藏文、蒙文、维吾尔文等主要的少数民族文字,为推进少数民族的信息化奠定了坚实的基础。

GB 18030—2000 的编码是在 GB 2312 和 GBK 的基础上进行的,它增加了四字节的编码,能完全映射 UCS/Unicode 的基本平面和辅助平面中的字符集。在此基础上,它包含的汉字数目增加到 27 000 多个,能适应出版、邮政、户政、金融、地理信息系统等迫切需要解决的人名、地名用字问题。GB 18030 代码空间的分配及码位的数目如表 1 所示。

GB 18030 编码空间约为 160 万码位,目前已编码的字符约 2.6 万。随着我国汉字整理和编码研究工作的不断深入,以及国际标准 ISO/IEC 10646 的不断发展,GB 18030 所收录的字符还会在新版本中逐步增加。

参考文献

1. GB 18030—2000《信息交换用汉字编码字符集·扩充集》
2. ISO/IEC 10646—1; Universal Multi - octet Character Set—UCS - Part 1: Architecture and Basic Multilingual Plane (张福炎)

xinxi xitong anquan pinggu zhunze

信息安全评估准则 (security evaluation criteria of information system)

根据计算机信息系统安全技术发展要求提出的信息系统安全保护等级划分和评估的基本准则。主要有:

(1) 可信计算机系统评估准则(TCSEC)

由美国国家计算机安全中心(NCSC)于1983年制订的计算机系统安全等级划分的基本准则,又称“橘皮书”(参见可信计算机系统评估准则)。1987年NCSC又发布了“红皮书”,即可信网络指南(TNI),提出了关于在局域网和广域网环境下的连网准则。1991年又发布了可信数据库指南(TDI)。

(2) 信息技术安全评估准则(ITSEC)

由欧洲四国(荷兰、法国、英国、德国)于1989年联合提出,俗称“白皮书”。在吸收TCSEC的成功经验基础上,首次在评估准则中提出了信息安全的保密性、完整性、可用性的概念,把可信计算机的概念提高到可信信息技术的高度。

(3) 通用安全评估准则(CC)

由美国国家标准技术研究所(NIST)、美国国家安全局(NSA)、欧洲的荷兰、法国、德国、英国以及加拿大等六国七方联合提出,并于1991年宣布,1995年发布正式文件。它的基础是欧洲的“白皮书”ITSEC、美国的包括“橘皮书”TCSEC在内的新的联邦评价准则、加拿大的CTCPEC以及国际标准化组织的ISO/SCITWGS的安全评价标准。

(4) 计算机信息系统安全保护等级划分准则

中国国家质量技术监督局于1999年发布国家标准,序号为GB 17859—1999(参见计算机信息系统安全保护等级划分准则)。

评价准则的出现为评价、开发、研究计算机及其网络系统的安全提供了指导准则。(胡道元)

xinxi yinbi

信息隐蔽 (information hiding)

在大型程序设计中,为了实现对象的可见性控制,在分层构造软件模块时要求有些对象只在模块内部可见,在该模块外部不可见的软件开发的原则和方法。例如在自顶向下分层设计中,其较低层的设计细节都被“隐蔽”起来,不仅功能的执行机制被隐蔽起来,而且控制流程的细节和一些数据也被隐蔽起来,随着设计逐步往低层推移,其细节也逐步显露出来。

在模块化设计中,接口只是功能描述,而模块本

身的实现细节对外界则是不可见的,实现上的改变并不影响使用它的模块,这有利于软件的重复使用。

例如,在Ada语言中,通过把包块分为两部分,使得外部元素和隐蔽元素明显地分开,包块式(又称包块规约)只说明外部元素,包块体则包括了隐蔽元素的说明,隐蔽元素只是支持外部元素的实现。

参考文献

徐家福. 系统程序设计语言. 北京: 科学出版社, 1983 (程虎)

xinyuan jiaohuan

信元交换 (cell switching) 以称为信元的定长分组为数据传输单位的一种交换技术。又称信元中继,它是异步传送模式(ATM)采用的交换技术。

(史美林)

xingshi fangfa

形式方法 (formal method) 建立在严格数学基础上的软件开发方法。在软件开发的全过程中,从需求分析、规约、设计、编程、系统集成、测试、文档生成、直至维护的各个阶段,凡是采用严格的数学语言,具有精确语义的方法,都属于形式方法。

在软件设计与研制中采用形式方法,可追溯到20世纪50年代后期对于程序设计语言编译技术的研究。那时J. Backus提出了BNF记法作为描述程序设计语言语法的元语言,出现了各种语法分析程序自动生成器以及语法制导的编译方法,使编译系统的开发从“手工艺制作方式”发展成具有牢靠理论基础的系统方法。

形式方法的研究真正形成热潮是在60年代后期。面对当时出现的所谓“软件危机”,人们提出了种种解决方法,归纳起来有两类:一是采用工程方法来组织、管理软件的开发过程,二是深入探讨程序和程序开发过程的规律,建立严密的理论,以期能用来指导软件开发实践。前者导致“软件工程”的出现和发展,后者则推动了对形式方法的深入研究。

形式方法所要解决的核心问题是程序的正确性。当今的软件系统极为庞大且高度复杂,不可能只靠测试来保证其正确性(荷兰的计算机科学家E. W. Dijkstra曾经有一句名言:“测试只能表明程序中存在错误,而不能表明程序中没有错误”)。因此,应用数学工具来加深对程序和程序开发过程

的理解,是从根本上解决程序正确性问题的一个重要途径。

形式方法的一个重要研究内容是**形式规约**,即用具有精确语义的形式语言书写的程序功能描述。形式规约之所以重要,是因为它是设计和编制程序的出发点,也是论证程序是否正确的依据。程序的正确性是一个相对的概念。说一个程序是否正确,总是相对于某个给定的规约而言。如果规约本身是非形式的、不精确的,就无从谈起程序的正确性。

一旦给出系统的形式规约,就可以推断它是否具有某种性质(比如不会发生死锁)。这些性质虽不是规约的一部分,但由于规约是形式的,所以可以在相应的形式推理系统中进行推导。如果用 SPEC, P 和 SYS 分别表示规约、性质和程序,用 \vdash 和 \models 分别表示推导和满足关系,并假定所用的推理系统是可靠的,那么从 $\text{SPEC} \vdash P$ 与 $\text{SYS} \models \text{SPEC}$ 就知道 $\text{SYS} \models P$ 。换句话说,只要规约具有性质 P,则任何正确地实现了该规约的程序都满足 P。注意这种推理与具体实现无关,可在动手编写程序之前进行。如果发现问题只须修改规约,从而避免了代价高昂的事后修改程序。

程序验证可以说是形式规约的孪生兄弟。事实上,能像证明数学定理那样,形式地证明程序的正确性,是提出形式规约的主要目的之一。每一种形式规约方法均伴随着相应的程序验证方法。比如用于推导命令式程序正确性的 Hoare 逻辑是建立在前、后断言规约方法上的;用于验证并发系统是否具有所希望性质的模型检查方法则基于时序逻辑。

围绕各种程序验证方法出现了大量的程序验证工具,既有自动的,也有交互式的。自动验证工具的好处是不需要人的参与,只要输入规约和程序,系统按照一定的算法,检查程序是否满足规约;在不满足时,大多数系统能给出诊断信息,指出不满足的原因。自动工具的局限性在于所能处理的问题类受限制。自动工具所能对付的问题必须是可判定的;即使对可判定的问题,如果时间、空间复杂性太高,在实际中也是不可行的。在交互式验证系统中,整个证明过程是在人的指导下进行的。系统可随时求助于人的智慧来决定下一步怎么做,因此从理论上说可以对付任何问题。但过多地要求人的干预会导致证明效率低下等问题。实际上大多数交互式工具都含有不同程度的自动化成分。

通过事后验证来保证程序的正确性,显得过于消极。不但难以措手,而且为时已晚。较为积极的

办法是研究程序开发过程本身,探寻保证正确性的程序开发方法。这些方法一般是“目标制导”的,即从给定的形式规约出发,应用适当的规则,逐步推出可执行的程序。由于这些规则是保持正确性的,所得到的程序一定满足所给的规约,而无须事后另作验证。这样的程序推导(或转换)涉及到不同层次上,即从规约到程序的各种对象,因而要求对所用的规约语言与编程语言的语义在同一框架中给出。例如在“最弱前置条件演算”中,规约断言与程序语句(“卫式命令”)的语义都统一用最弱前置条件来定义。程序推导与转换在函数式与逻辑式程序设计风范中也得到深入的研究。

在直觉主义逻辑中,“证明”必须是能行的,利用最近 20 年来直觉主义类型论的成果,可以将类型论中的“谓词”看作对程序的规约,谓词为真的证明即是(带类型的) λ -演算的项,因而可看作是函数式程序语言中的程序。这样,“程序满足规约”这句话翻译为类型论的语言就是“论据证明谓词为真”。基于这种对应关系,就可以在直觉主义逻辑系统中进行证明,而从证明中“抽取”出程序(参见**基于类型理论的方法**)。

近年来对实时系统形式方法的研究取得了很大进展。在实时系统中时间因素对系统的行为起关键的甚至是决定性的作用。典型的实时系统有飞机或导弹的飞行控制系统、各类工业过程控制器以及通信系统等。目前比较成功的用于实时系统的方法,多数是在已有的关于非实时系统的方法基础上扩充时间因素,比如在时序逻辑(线性或区间)中引入时间参数,或在传统的进程代数中增加表示时间的算子。一般要求这种扩充是保守的,即在原形式系统中为真的事实在扩充后的系统中仍然为真。**混成系统**是一类特殊的实时系统,其特点是既随时间而连续变化,又受离散突发事件的驱动。对混成系统形式方法的研究方兴未艾。

形式方法的另一个重要应用领域是系统的分解与集成。大型软件系统由成千上万个模块组成,这些模块又按一定的原则组织成一些子系统。模块和子系统的对外界面与版本信息,以及模块、子系统之间的调用、联结关系,都可以形式地加以描述。这种系统结构描述一般在系统设计阶段进行。在系统集成阶段可以利用这种结构描述进行界面与版本一致性检查,也可以自动地从模块库产生最终的可运行系统。实际上,几乎所有的软件开发环境都不同程度地提供这方面的支持。

形式方法的研究用到逻辑、代数、范畴论诸数学分支的成果,同时反过来也推动了这些数学分支的发展。比如 λ -演算、时序逻辑、直觉主义类型论近30年来取得的重大进展,都与形式方法有直接的关系。

经过近30年的大量、艰辛的研究,人们在形式方法这一领域已经取得了许多重要的成果,并应用于实际。比如对于抽象数据类型的讨论为高级语言中模块构造的设计提供了理论基础,也是产生面向对象程序设计风范的源头之一;对于通信并发系统的理论研究,导致了用于书写通信协议规约的国际标准语言 Estelle 和 Lotos 的诞生;程序验证系统查出了软件和硬件设计中的许多错误和隐患;现有的自动验证技术已能对付具有 10^{300} 以上状态的系统。但从总体看,迄今为止形式方法所能处理的问题的规模还嫌太小。如何将已经成功地应用于中小规模问题的方法“放大”以解决真正工业规模的问题,是形式方法当前所面临的一个重大挑战。

参考文献

1. Wing J M. A Specifier's Introduction to Formal Methods. IEEE Computer, 1990, 23(9): 8~24
2. Leveson N G ed. Special Issue on Formal Methods in Software Engineering. IEEE Transactions on Software Engineering, 1990, 16(9) (林惠民)

xingshi gongneng guiyue

形式功能规约 (formal functional specification) 用具有坚实数学基础的语言书写的软件功能规约,即一个合乎形式功能规约语言语法与语义的陈述。如果这个陈述无矛盾,相对于需求完备,并且无歧义,则该规约有效。对一个有效的规约可以有多个实现版本,一个正确的实现是指它是该规约的一个模型,即是该规约的一种语义解释。

在形式功能规约中,一个功能的描述有两部分:型构和制约关系。型构指出该功能的使用方式,制约关系指出该功能的行为特征。根据制约关系,形式功能规约可分成模型抽象规约和性质抽象规约。采用模型抽象方法,规约设计者通过直接构造系统的数学模型结构来描述系统的行为,它除了描述功能以外,还可描述功能分解结构。采用性质抽象的方法则用一组公理限定系统的行为,系统必须满足所有的公理,因此公理越少,可能的实现就越多。根据公理形式的不同,性质抽象方法又可分成两类,基于 Hoare 逻辑的公理化方法和基于抽象数据类型的

代数方法。前者主要用于过程抽象,理论基础是前后断言方法及最弱前置条件方法。后者主要用于数据抽象,以异调代数为理论基础,公理被限制为等式。支持上述规约方法的语言请参见功能性语言。

与非形式功能规约相比,形式功能规约有利于分析规约的一致性和完备性,易于保证无歧义。也有利于程序的开发和正确性验证。经验表明,对具有一定数学基础的人来说,形式功能规约较非形式功能规约更易于理解。(伊波)

xingshi guiyue

形式规约 (formal specification) 用具有坚实数学基础的语言和方法给出的软件的功能描述或设计描述。软件的规约既指对最终产品的功能描述、实时特性、运行效率及安装与运行的时空要求等等,也指对开发过程的各种要求,比如交货期限、工程预算等等,亦可指软件设计描述。它是软件设计与研制工作的出发点,也是衡量最终产品是否合格的依据。形式规约只关心软件产品的功能性质,即“做什么”,而不关心其功能是怎样实现的。

形式规约的最主要特点是具有严格的数学基础,因此有可能讨论它的一致性与完备性等性质。一致性指的是自身无矛盾,这是任何规约都应满足的基本要求。自相矛盾的规约在理论上是没有任何意义的,在实践上是有害的。因为它将导致人力与财力的浪费。完备性是指规约是否完全、无遗漏地刻画了它所描述的对象。完备的规约具有许多良好的性质,比如语义模型的惟一性等。

形式规约的一个重要应用是程序正确性验证。只要所用的编程语言同样具有严格的数学语义,就可以形式地论证程序是否满足所给的规约。如果满足,就说明程序具有规约所刻画的性质。也就是说,程序相对于该规约是正确的。验证程序的正确性需要进行大量的推理和计算,因此出现了许多支持程序正确性验证的软件工具,即程序证明器。这些工具既有自动的,也有交互式的。

按所依据的数学理论,形式规约可以分为基于集合论的,基于逻辑的与基于代数的;按所适用的程序类,又可以分为面向顺序程序的与面向并发程序的。以下对常用的形式规约方法作简要的介绍。

前后断言方法可以说是最早的形式规约方法,出现于20世纪60年代后期,首倡者是 C. A. R. Hoare。其基本思想是在程序语句间插入刻画程序状态(即程序变量的取值)的一阶谓词公式,称为断言。插

在语句之前的断言指明执行此语句之前程序变量所应具有的性质,称为前断言;插在语句之后的断言则刻画了语句执行结束时的程序状态空间,称为后断言。后断言可以看作是对程序所应实现的任务的描述;前断言则可看作是正确执行程序的前提。一对前、后断言即是对程序的规约。如果用 S, P 和 Q 分别表示程序、前断言和后断言,则程序 S 满足由 P, Q 组成的规约这一事实可用三元组 $P \vdash S \vdash Q$ 来表示。C. A. R. Hoare 提出了一套公理系统用于推导这样的三元组,如果 $P \vdash S \vdash Q$ 可由该系统推出,则 S 就是相对于 P, Q 正确的。考虑到程序是否终止,这种正确性又可分为部分正确性与完全正确性两类。

形式规约是对程序“做什么”的数学描述,这种描述主要体现在后断言中,因此后断言往往包含了关于程序结构的重要信息。基于这种认识, E. W. Dijkstra 提出了一个推导正确程序的形式系统,称为**最弱前置条件演算**。其基本思想是将程序设计看作是“面向目标”的活动,编程就是从预先给定的后断言出发,逐步推导出满足它的程序,同时计算出所需的最弱前置条件。

前、后断言法主要面向程序的控制结构,而程序是控制结构和数据结构的统一体。这一事实在 70 年代初兴起的研究抽象数据类型的热潮中得到了生动的反映。抽象数据类型由一组数据和施加于该数据上的运算组成。对于这组数据的操作只能通过在该抽象数据类型中定义的运算进行,从而达到了数据抽象和信息隐藏的目的。

为了将前、后断言法推广应用于抽象数据类型的规约,需要引入“表示类型”与“数据不变式”的概念。所谓“表示类型”即在规约中用来表示抽象类型的具体数据类型。比如先进后出栈可用数组加指针来表示。作为表示类型的数据类型的语义必须是已知的;它们可以是在规约语言中预定义的类型,也可以是已经给出了形式规约的其他抽象数据类型。由于抽象数据类型的数据空间常常只是表示类型的一个子集,需要用一谓词来对表示类型加以限定,称为数据不变式。表示类型的元素中只有满足数据不变式的才是抽象类型的合法元素。给定表示类型与数据不变式后,抽象类型上的运算可用前后断言来刻画,这些断言中可引用表示类型上的运算。这种方法称为“面向模型的方法”,主要用在 VDM 和 Z 中。

抽象数据类型的另一种(也是更为流行的)规约方法是代数方法,通常称为代数规约。其基本出

发点是用基调给出抽象数据类型各运算的类型,而用一组代数公理来规定这些运算所应具有的行为。代数规约的语义即是满足该公理集的某类模型(参见代数规约)。代数方法的长处一是数学基础严密,建立在传统的泛代数理论上;二是抽象程度高,只涉及性质,无须诉诸数据的表示,因此又称为面向性质的方法。

在书写规约时应注意的一个问题是如何描述得恰如其分,既不过多也不过少。在规约中描述过多会导致“实现偏向”,给实现施加了不必要的限制,从而排除了一些原本是合理的实现;描述得过少又有容纳不合理实现的危险。为了使规约写得恰到好处,除了应透彻理解、熟练掌握所使用的规约语言和方法外,更重要的是对所要描述的系统有全面深入的了解。

近年来有些学者研究基于直觉主义类型理论的形式规约方法。这种方法将规约看成谓词(类型),将实现规约的程序看成该谓词为真的证明。其特点是从规约的正确性证明中抽取可执行程序,这个程序就是该规约的一个实现。

并发程序比顺序程序表现出更为复杂的行为,对它们的规约也更为困难。

时序逻辑是描述并发程序性质的一个有力工具。并发程序的一些重要性质,比如活性(“好的”事件终将发生)、安全性(“坏的”事件永远不会发生)都可以用时序逻辑公式表达。通过在程序中插时序断言,可用与前后断言法类似的方法来证明一个并发程序满足某时序逻辑公式。对于有限状态的并发系统还可以用模型检查算法来自动判定是否具有用某类时序逻辑公式表达的性质。

70 年代末英国学者 C. A. R. Hoare 与 R. Milner 分别提出了用于描述通信并发系统的代数语言 CSP 与 CCS,后来发展成进程代数理论。进程代数研究的核心问题之一是等价性,即如何决定两个语法上不同的进程表达式表示同一个语义对象。如果用某进程表达式作为一并发系统的规约,则另一与之等价的且包含更多并发性的表达式就可以看作是该规约的实现。进程代数理论提供了多种方法来论证这种实现的正确性。

形式规约研究的重要内容之一是形式规约语言,即用于书写形式规约的语言。不同的形式规约方法要求不同的形式规约语言。比如 Z 和 VDM 各有其规约语言。代数规约语言就更多了,如 Clear, ASL, ACT-ONE / TWO, OBJ。还有面向分布式系统

的 Estelle 和 Lotos 等。由于所依据的数学理论及规约方法不同,各种规约语言千差万别。但从结构上看,却具有一个显著的共性:每个规约语言都由两部分组成:描写基本规约(或原子规约)的成分和把基本规约组合成大规模规约的构造成分。其中后者是形式规约语言研究和设计的重点,也是衡量规约语言优劣的主要依据。常见的构造成分有:包含、并、交、扩充、参数化等。

在整个软件设计和开发过程中,形式规约是从非形式化的、不精确的现实世界到形式化的、精确的程序世界的分界岭。如何从非形式需求得到形式化的规约?这就是形式规约的获取问题,是这一领域中的一个重要研究课题。

参考文献

Wing J M. A Specifier's Introduction to Formal Methods. IEEE Computer, 1990, 23(9): 8~24

(林惠民)

xingshi sheji guiye

形式设计规约 (formal design specification)

参见设计规约。

xingshi yuyan lilun

形式语言理论 (formal language theory)

用数学方法研究自然语言(如英语)和人工语言(如程序设计语言)的语法的理论。它只研究语言的组成规则,不研究语言的含义。

形式语言理论在自然语言的理解和翻译、计算机语言的描述和编译、社会和自然现象的模拟、语法制导的模式识别等方面有广泛的应用。

形式语言的研究始于 20 世纪初,50 年代中期将形式语言用于描述自然语言。1956 年, N. Chomsky 发表了用形式语言方法研究自然语言的第一篇文章。他对语言的定义方法是:给定一组符号(一般是有限多个),称为字母表,以 Σ 表之。又以 Σ^* 表示由 Σ 中字母组成的所有有限长符号串(或称字,包括空字,也称句子)的集合,则 Σ^* 的每个子集都是 Σ 上的一个语言。例如,若令 Σ 为 26 个拉丁字母加上空格和标点符号,则每个英语句子都是 Σ^* 中的一个元素,所有合法的英语句子的集合是 Σ 上的一个语言。乔姆斯基的语言定义方法为人们所公认,一直沿用下来。

1960 年,算法语言 ALGOL 60 报告发表,其中第

一次使用一种称为巴克斯范式的方法来描述程序设计语言的语法。不久,人们即发现巴克斯范式系统极其类似于形式语言理论中的上下文无关文法,从而打开了形式语言广泛应用于描述程序设计语言的局面,使它发展成为理论计算机科学的一个重要分支。

形式语言和文法 形式语言理论以无限的语言为主要研究对象。例如,所有以 n 个 a 构成的字 ($n \geq 1$) 组成一个语言 $L_a = \{a, aa, aaa, \dots\}$, 它就是无限的。因此,研究形式语言遇到的第一个问题就是描述问题。描述的手段必须是严格的,而且必须能以有限的手段描述无限的语言。乔姆斯基用变换文法作为形式语言的描述手段。例如, L_a 可用如下的变换文法描述: $\{S \rightarrow a, S \rightarrow aS\}$ 。这个文法由两条变换规则组成。每一步变换(也叫推导)都用一条变换规则的右部替换它的左部。例如,句子 $aaaaa$ 可以这样推导出来: $S \rightarrow aS \rightarrow aaS \rightarrow aaas \rightarrow aaaaaS \rightarrow aaaaa$ 。

严格地说,变换文法定义成四元组 $G = (\Sigma, V, S, P)$ 。 Σ 是字母表,又称终结符号表。 V 是变量表,又称非终结符号表; S 是出发符号; P 是变换规则(又称产生式)的集合,其中 Σ, V 和 P 都是有限集, $\Sigma \cap V = \emptyset, S \in V$ 。又令 α 和 β 分别表示 $(\Sigma \cup V)^+ - \Sigma^+$ 和 $(\Sigma \cup V)^*$ 中的元素(用 \cdot 代替 $*$ 表示不含空字),则 P 中所有产生式皆形如 $\alpha \rightarrow \beta$ 。这样定义的文法称为 0 型文法,又称短语结构文法,所生成的语言称 0 型语言。每个 0 型语言都是递归可枚举集(参见图灵机),反之亦然。

以 $|\alpha|$ 表示符号串 α 的长度。对 0 型文法加限制 $|\alpha| \leq |\beta|$, 即得到 1 型文法,生成的语言称 1 型语言。1 型文法也可以这样定义:它的所有产生式均取 $\gamma A \delta \rightarrow \gamma \omega \delta$ 的形式,其中 $\gamma, \omega, \delta \in (V \cup \Sigma)^*, |\omega| > 0, A \in V$ 。其直观意义是:在左有 γ , 右有 δ 的环境下, A 可以被 ω 替换。因此,1 型文法和 1 型语言又分别叫上下文有关文法和上下文有关语言。

如果要求 0 型文法中 $\alpha \in V$, 便得到 2 型文法,又称上下文无关文法,生成的语言称 2 型语言或上下文无关语言。不含空字 ε 的 2 型语言必可由不含空产生式 $A \rightarrow \varepsilon$ 的 2 型文法生成,其中 $A \in V, \varepsilon$ 在书写时可省去。

若要求 2 型文法中产生式的右端为 aB 或 a , 其中 $a \in \Sigma, B \in V$, 则得到 3 型文法,或称正则文法,又称右线性文法,生成的语言称为 3 型语言,或正则语言,或右线性语言。

以 G_0, G_1, G_2, G_3 分别代表上述四类文法, 以 L_0, L_1, L_2, L_3 分别代表四类语言 (其中 L_2 不含空字), 又以 L_i 表示由递归集构成的语言类, 则有 $L_3 \subset L_2 \subset L_1 \subset L_0$, 这些包含都是真包含。例如, 取 $\Sigma = \{a, b, c\}$, $L_0 = \left\{ \bigcup_{i=0}^{\infty} x_i \mid x_i \text{ 为任意一个由第 } i \text{ 台图灵机接受的语句} \right\}$ 是零型语言而不是递归集。 $L_1 = \left\{ \bigcup_{i=0}^{\infty} x_i \mid x_i \text{ 为任意一个不能由第 } i \text{ 个 1 型文法产生的语句} \right\}$ 是递归集而不是 1 型语言。 $L_2 = \{a^n b^n c^n \mid n \geq 1\}$ 是 1 型而非 2 型语言。 $L_3 = \{a^n b^n \mid n \geq 1\}$ 是 2 型而非 3 型语言。 $L_3 = \{a^n \mid n \geq 1\}$ 是 3 型语言, 这里 a^n 表示 n 个 a 的连接。

形式语言和自动机 上述文法和语言的分层方法, 是乔姆斯基于 1959 年提出来的, 因而称为乔姆斯基分层。这种分层法提出不久, 人们即发现它和自动机的分类有密切的关系。到 1964 年, 四类文法及其语言全部在自动机中找到了它们所对应的位置。0 型、1 型、2 型和 3 型语言正好分别是图灵机、非确定型线性有界自动机、非确定型下推自动机和有限自动机接受的语言。确定型和非确定型图灵机接受的语言是相同的, 确定型和非确定型有限自动机接受的语言也是相同的。确定型下推自动机接受的语言集合是三型语言的真子集, 称为确定型上下文无关语言。至于非确定型线性有界自动机和确定型线性有界自动机接受的语言集合是否相同, 是一个著名的尚未解决的问题, 简称 **LBA 问题**。

形式语言的代数性质 研究形式语言的第三种途径是把它作为一种代数结构来考察。可以施行于语言的代数运算, 包括求交 ($L_1 \cap L_2$)、求并 ($L_1 \cup L_2$)、求差 ($L_1 - L_2$)、求补 ($\sim L$, 即 $\Sigma^* - L$)、反演 (L^{-1} , $ab \in L \leftrightarrow ba \in L^{-1}$)、乘积 ($L_1 \cdot L_2, W_1 \in L_1, W_2 \in L_2 \rightarrow W_1 W_2 \in L_1 \cdot L_2$)、乘幂闭包 ($L^{\circ}, W \in L \rightarrow W^n \in L^{\circ}, n \geq 0$)、无空字乘幂闭包 (L° 减去空字 ε)、同态 ($L_1 \rightarrow L_2$)、无空字同态、置换 ($f(L), L$ 中每个字母置换成一个语言、字母串置换成与串中各字母对应之语言的乘积)、正则置换 (置换语言都是正则语言)、 L_1 对 L_2 左商 ($L_2 \setminus L_1 = \{Q \mid PQ \in L_1, P \in L_2\}$)、右商 (L_1 / L_2) 等。早在 1961 年即有人指出: 一个上下文无关语言和一个正则语言之交仍是上下文无关语言。不久人们发现, 这一结果具有相当的普遍性: 几乎所有重要的语言族都具有在与正则语言相交下

封闭的性质。从 60 年代中期开始, 研究某些语言族在某些代数运算下的封闭性已经成为形式语言代数研究的一个中心课题。乔姆斯基分层中各型语言在一些代数运算下的封闭性如表 1 所示。

表 1 四类语言在代数运算下的封闭性

代数运算 \ 语言族	L_0	L_1	L_2	L_3
求并	✓	✓	✓	✓
乘积	✓	✓	✓	✓
乘幂闭包	✓	✓	✓	✓
无空字乘幂闭包	✓	✓	✓	✓
求补	×	?	×	✓
求交	✓	✓	×	✓
与正则语言相交	✓	✓	✓	✓
反演	✓	✓	✓	✓
置换	✓	×	✓	✓
无空字置换	✓	✓	✓	✓
同态	✓	×	✓	✓
无空字同态	✓	✓	✓	✓
对正则语言的左商	✓	×	✓	✓
对正则语言的右商	✓	×	✓	✓
正则置换	✓	×	✓	✓
无空字正则置换	✓	✓	✓	✓

抽象语言族 简称 AFL, 由 S. Ginsburg 等人于 1966 年提出, 是研究语言族在代数运算下封闭性质的有力工具。令 Σ_* 为无限字母表, 在其任一有限子集 Σ_i 上构造语言 $L_i \subseteq \Sigma_i^*$ 。如果任何一组语言 $\{L_i\}$ 中至少包含一个 $L_{i_0} \neq \emptyset$, 则称 $\{L_i\}$ 为一语言族。在同态、逆同态和正则语言相交下保持封闭的语言族称为满三重组。对并运算封闭的满三重组称为满半 AFL。对乘幂闭包封闭的满半 AFL 称为满 AFL。从一个语言族 \mathcal{L} 出发, 经上述代数运算后得到的闭包分别称为由 \mathcal{L} 生成的满三重组、满半 AFL 和满 AFL, 以 $\hat{\mathcal{M}}(\mathcal{L})$ 、 $\hat{\mathcal{S}}(\mathcal{L})$ 和 $\hat{\mathcal{F}}(\mathcal{L})$ 表之。如果语言族 \mathcal{L} 只包含一个语言 L , 则由 \mathcal{L} 生成的结构分别称为满主三重组、满主半 AFL 及满主 AFL。如果把同态限制为无空字同态, 即不得把非空字映为空字, 则所有以上定义中的“满”字皆应除去。

一个语言族成为 AFL 的充分必要条件是它在并运算、无空字乘幂闭包、无空字正则置换、与正则语言相交及有界同态 (对固定的 $k \geq 1$, 任一语句的同态映象之长度不超过原长 k 倍) 下是封闭的。一个语言族成为满 AFL 的充分必要条件是它在并运算、乘幂闭包、正则置换、与正则语言相交及同态映

射下是封闭的。把乔姆斯基的四类语言看作语言族 $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$, 则它们都是 AFL, 其中 $\mathcal{L}_0, \mathcal{L}_2, \mathcal{L}_3$ 是满 AFL, \mathcal{L}_1 不是, 因为它在一般的同态映射下不封闭。

判定问题 一些已有的成果如表 2。表中 D 表示可判定, U 表示不可判定, G 表示文法, L 表示语言。

表 2 与形式语言有关的判定问题

判定问题 \ 语言族	L_0	L_1	L_2	L_3
任意字 $x \in L(G)$?	U	D	D	D
$L(G_1) \subset L(G_2)$?	U	U	U	D
$L(G_1) = L(G_2)$?	U	U	U	D
$L(G) = \emptyset$?	U	U	D	D
$L(G)$ 为无限集合?	U	U	D	D
$L(G) = \Sigma^*$?	U	U	U	D
$L(G) \in L_3$?	U	U	U	/
$L(G) \in L_2$?	U	U	/	/
G 无二义?	/	/	U	D
$L(G)$ 无二义?	/	/	U	/
$\sim L(G) = \emptyset$?	U	U	U	D
$\sim L(G)$ 为无限集合?	U	U	U	D
$\sim L(G) \in L_3$?	U	U	U	/
$\sim L(G) \in L_2$?	U	U	U	/
$\sim L(G)$ 和 $L(G)$ 同一类型?	U	?	U	/
$L(G_1) \cap L(G_2) = \emptyset$?	U	U	U	D
$L(G_1) \cap L(G_2)$ 为无限集合?	U	U	U	D
$L(G_1) \cap L(G_2) \in L_3$?	U	U	U	/
$L(G_1) \cap L(G_2) \in L_2$?	U	U	U	/

非乔姆斯基的文法和语言类 基本原理是对每一步变换时允许使用的产生式加以不同的限制。例如, 矩阵文法把产生式分成有限多个有序组, 要求每一步变换必须连续地使用整组产生式。时控文法也把产生式分组, 规定每一时刻 t 只能从第 t 组中选一产生式加以应用。以 $\varphi(t)$ 表示第 t 组产生式, 则当有 t_0 使 $\varphi(t+t_0) = \varphi(t)$ 时, 它称为周期时控文法。有序文法把产生式排成偏序, 只有当排在前面的产生式不能应用时, 才允许使用后面的产生式。Lindenmeyer 于 1968 年提出的 L 系统以细胞自动机和生物发育模型为背景。它规定在每一步变换中, 对同样的左部符号必须使用同一个产生式。例如若 $B \rightarrow \alpha$ 和 $B \rightarrow \beta$ 同为产生式, 则从 BAB 只能推出 $\alpha A \alpha$ 和 $\beta A \beta$, 而不能推出 $\alpha A \beta$ 或 $\beta A \alpha$ 。 L 系统的出发符号是一个字母串, 一般没有终结符或非终结符之分。在名称上, 它用 0 表示零面(产生式上下文无关), I

表示上下文有关, D 表示决定型, P 表示增值型(不产生空字), T 表示产生式按表格分组。例如, $DTOL$ 语言表示由决定型、表格型、零面 L 系统生成的语言。

ω 语言 令 $\Sigma^\omega = \left\{ \prod_{i=1}^{\infty} a_i \mid a_i \in \Sigma \right\}$, 其中 Σ 为

有限字母表, 则 Σ^ω 的元素称为 ω 字, Σ^ω 的子集称为 Σ 上的 ω 语言。设 A 是以 Σ 为输入字母表的有限自动机, 则 $(A, F) = A^\omega$ 称为 ω 有限自动机(ωFSA), 其中 $F \subseteq 2^K$, K 是 A 的状态集。令 $\prod_{i=1}^{\infty} S_i$ 为 A^ω 在识别

ω 字 α 过程中经历的状态序列, 则

1. $A^\omega 1$ 型接受 α , 若 $(\exists H \in F)(\exists i) S_i \in H$
2. $A^\omega 1'$ 型接受 α , 若 $(\exists H \in F)(\forall i) S_i \in H$
3. $A^\omega 2$ 型接受 α , 若 $(\exists H \in F)(\forall k)(\exists i) S_i^k \in H$
4. $A^\omega 2'$ 型接受 α , 若 $(\exists H \in F)(\forall k)(\forall i) S_i^k \in H$
5. $A^\omega 3$ 型接受 α , 若 $\{S_i^k \mid i, k \text{ 任意}\} \in F$

其中 S_i^k 是第 k 种在 $\prod_{i=1}^{\infty} S_i$ 中无穷次出现的状态, $S_i^k = S_i$ 。

麦克纳登证明了 ωFSA 的 2 型和 3 型接受以及 $\omega DFSA$ 的 3 型接受都是等价的(D 表示确定型), 并以此定义 ω 正则语言。 ω 正则语言在所有布尔运算下封闭。对任意 ω 正则语言, 下列问题可判定: ① L_1 为空、有限或无限, ② $L_1 = L_2$, ③ $L_1 \subseteq L_2$, ④ $L_1 \cap L_2 = \emptyset$ 。

科恩等人用类似方法定义 ω 下推自动机 ωPDA , 还定义了 $\omega 3$ 型文法 ωRG 和 $\omega 2$ 型文法 ωCFG , 证明了 ωRG 和 ωCFG 生成的语言正好被 ωFSA 和 ωPDA 接受。

高维文法 上面讨论的文法只许 Σ 含简单字母, 称为串文法。若允许 Σ 为树或图的集合, 即是树文法和图文法, 生成的语言称为树语言和图语言。这些文法统称高维文法, 也有人研究更复杂的高维文法。

参考文献

1. Revesz G E. Introduction to Formal Languages. New York; McGraw-Hill, 1983
2. Book R V. Formal Language Theory Perspectives and Open Problems. New York; Academic Press, 1980

(陆汝铃)

xingshi yuyi

形式语义 (formal semantics) 用数学方法,尤其是形式系统严格定义出的语言的语义。程序设计语言是人们用来和计算机系统进行通信和控制其工作的人工语言。作为语言,人工语言和自然语言(如汉语、英语等)一样,有其语法、语义和语用范畴。程序设计语言的语法是指程序的组成规则,语义是指程序的含义。

程序设计语言的语义通常是由设计者用一种自然语言非形式地解释的,实现者和使用者则依据各自的理解去实现和使用这种语言。然而使用自然语言和非形式的方法解释语义,容易产生歧义,造成语言设计者、用户和实现者对语义的不同理解,影响语言的正确实施和有效使用。程序设计语言中的过程调用语句就是这方面的一个典型例子。人们发现对过程调用语句的非形式解释可能导致各种不同的理解,产生多种不同的效果。

为了正确、有效地使用程序设计语言,必须了解语言中各个成分的含义,并且要求计算机系统执行这些成分所产生的效果与其含义完全一致。人们这种对语义精确解释的要求便产生了形式语义学。形式语义学的研究始于 20 世纪 60 年代初期,在程序设计语言 ALGOL 60 的设计中,第一次明确区分了语法的语法和语义,并使用 BNF 符号系统成功地实现了语法的形式描述。语法的形式化大大推动了语义形式化的研究,围绕 ALGOL 60 的语义出现了形式语义学早期的研究热潮。以后的程序设计语言,如 PASCAL, Ada 等,都有人给出了严格的形式语义,旨在为编制程序语言的编译程序提供正确依据。

美国斯坦福大学 J. McCarthy 于 1962 年系统地论述了程序设计语言语义形式化的重要性,以及它同程序的正确性、语言的正确实现等的关系,并提出在形式语义研究中使用抽象语法和状态向量等方法。近年来,形式语义的理论和应用都有了很大发展。

程序设计语言的语法是规定程序组成方法的一些规则,称为具体语法,但在定义程序的语义时,必须首先识别给定的程序,分析程序的语法结构。因此,在形式语义中使用一种讨论程序分解的语法规则,这种语法称作抽象语法。不同的程序设计语言往往使用不同的记号和表示方式。形式语义提供的方法适用于一切程序设计语言,故抽象语法采用的记号和表示方式也是具体语法的一种抽象。

在定义程序设计语言的语义时,需要一种定义语义的语言,这种语言称为元语言。元语言可以采用已有的数学语言,也可以是以数学理论为基础的专门设计的语言。用元语言去定义程序语言的形式语义,必须首先严格定义元语言的语义。

用程序设计语言编写的程序,规定了它对计算机系统中数据的一个加工过程,形式语义的基本方法是将程序加工数据的过程及其结果形式化,从而定义程序的语义。

由于形式化中侧重面和使用的数学工具不同,形式语义可分为四大类。①**操作语义**:着重模拟数据加工过程中计算机系统的操作;②**指称语义**:主要刻画数据加工的结果,而不是加工过程的细节;③**公理语义**:用公理化的方法描述程序对数据的加工;④**代数语义**:把程序设计语言看作是刻画数据和加工数据的一种抽象数据类型,使用研究抽象数据类型的代数方法,来描述程序设计语言的形式语义。

参考文献

周巢尘. 形式语义学引论. 长沙: 湖南科技出版社, 1985
(周巢尘 李晓山)

xuni cunchuqi

虚拟存储器 (virtual memory) 在具有层次结构存储器的计算机中,为用户提供比**主存储器**容量大得多的可随机访问的地址空间。虚拟存储技术使**辅助存储器**和**主存储器**密切配合,对用户来说,好像计算机具有一个容量比实际主存大得多的主存可供使用,因此称为虚拟存储器,简称虚存。虚拟存储器的地址称为**虚地址**或**逻辑地址**。程序运行时,中央处理机实际访问的存储器仍然是**主存储器**,主存和辅存称为**实际存储器**,简称**实存**。实存的地址称为**实地址**或**物理地址**。

虚拟存储器的概念于 1961 年由英国曼彻斯特大学提出,并在 Atlas 计算机中实现,称为一级存储器。Atlas 计算机的主存储器容量为 16 kB,外存储器(磁鼓存储器)的容量为 96 kB。主存分为 32 页,每页 512 个字。如果被访问的内容不在主存而在磁鼓存储器中,计算机自动将该内容所在的页从磁鼓存储器调入主存以替换主存中的某一页,并将被替换的页送回磁鼓存储器。主存和磁鼓存储器之间的页交换对用户是透明的。用户使用计算机时,好像随机存取存储器的容量很大,而且存取速度也较快。在现代计算机中,虚拟存储器已得到广泛应用,不但

用于大型计算机,也用于小型计算机和微型计算机。

虚拟存储器的虚实地址转换,块映射方式和块替换策略在原理上和高速缓冲存储器的类似。它们的主要不同之处是高速缓冲存储器的机制基本上由硬件实现,而虚拟存储器的机制由操作系统在硬件的配合下实现。硬件主要负责虚实地址转换,操作系统负责调页管理、实存管理、主存和辅存之间的信息调度等。

对于具有虚拟存储器的计算机,编译程序产生的程序是用虚地址编程的,程序由操作系统装入辅存中。程序运行时,虚地址被转换为实地址,如果所需的内容已在主存中,则中央处理器访问主存的实地址;如果所需的内容不在主存,称为页面失效,计算机产生页面失效中断,操作系统将要访问的那一页从辅存调入主存。

虚实地址转换

(1) 页式虚拟存储器的虚实地址转换 在页式虚拟存储器中,主存和辅存都划分为大小相同的页面,主存的页面按顺序编号,称为实页号。程序也划分为大小与页面相同的页,并按顺序编号,称为虚页号。程序中的页可以装在主存的不连续的任意页面中。虚页向实页的映射(即虚页号和实页号的对应关系)由页表给出。每一个装入计算机的程序(进程)都有自己的页表。图1为页表的简单例子。设主存中已装入A,B,C 3个进程,现在要调入辅存中的占有3个页面的进程D。操作系统将进程D的3个页装入主存中的3个不连续的空闲的页面,并建立进程D的页表。页表中的一行称为页表项。每一页表项有若干个控制位,其中1个位为装入位,它为1时表示该页已在主存中,为0时表示不在。根据页表可将虚地址转换为实地址。

页表是操作系统根据主存的运行情况自动建立的,对程序员透明。页表平时可以存放在辅存中,在需要时调入主存。在主存中有存放页表的页表区。每个页表都有自己的页表起始地址。在程序运行时,存储管理软件将该程序的页表起始地址送到页表基址寄存器,如图2所示,页表起始地址和虚地址中的虚页号拼接成页表地址,从而可在页表中找到实页号。如果页表项中的装入位为1,表示该页已调入主存储器,页面有效,可根据此实页号去访问主存;如果装入位为0,则表示该页未调入主存,产生缺页中断,操作系统将所需的页从辅存调入主存。

(2) 段式虚拟存储器的虚实地址转换 在段式

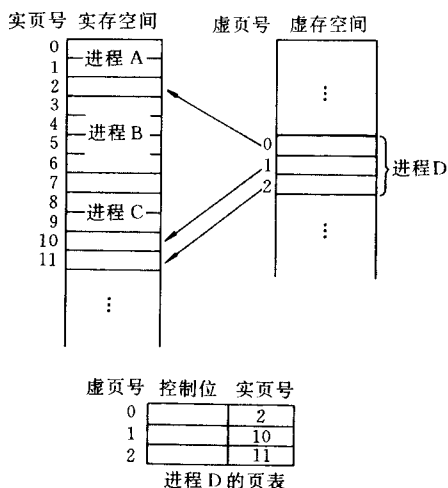


图1 页表举例

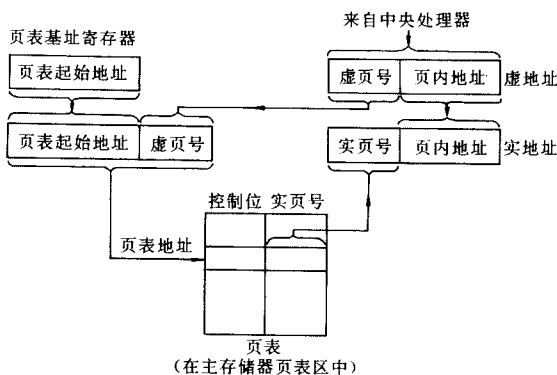


图2 页式虚拟存储器的虚实地址转换

虚拟存储器中,段是按照程序的逻辑结构划分的,段的长度因程序不同而不同。虚地址由段号和段内地址组成。为了将虚地址转换成实地址,需要一个段表。段表指明程序中的各个段在主存储器中的实际位置。凡装入计算机的每一个程序都有自己的段表。段表由段表项组成,每一段表项包含该段在主存中的起始地址、段的长度以及装入位等控制位。段表本身也是一个段,平时可存放在辅存中,需要时调入主存。和页式虚拟存储器相似,在访问存储器时,要先查阅段表,在段表中去寻找实地址。

(3) 段页式虚拟存储器的虚实地址转换 在段页式虚拟存储器中,主存分为页面,程序分为段,每个段又分为若干个和主存页面同样大小的页。虚地址包含段号、页号、页内地址等。凡装入计算机的每一个程序都有1个段表和1组页表(每一段有1个

页表)。段表项中有指明该段的页表在主存中的起始地址和页表长度等信息,每个段的页表则进一步指明该段中的各页在主存中的实际位置。段的起始地址不能是任意的,必须位于主存页面的起始单元中。每次访问存储器时,要先查段表,得到页表的起始地址后,再查页表,才能找到实地址。

在上述的3种虚实地址转换时,由于页表和段表都存放在主存中,需要事先访问主存一两次,从而使中央处理器访问存储器的时间加长。为了缓解这个问题,在虚拟存储器中,可增设**转换检测缓冲器**,它使虚实地址的转换加快。

(4) 虚地址到辅存实地址的转换 在缺页中断时,操作系统应将所需的虚页从辅存调入主存,调页时虚地址应转换成辅存实地址。现代计算机一般都采用磁盘存储器作为辅助存储器。磁盘的盘片表面上的磁道划分为若干个扇区,每个扇区的大小通常等于1个主存页面。磁盘的实地址由磁盘机号、柱面号、磁头号和扇区号组成。虚页号与磁盘实地址的对应表称为**外页表**。外页表也是按虚页号的顺序排列的。外页表项的格式示于图3。外页表的内容是在将程序装入辅存时填写的,其中M是装入位。M为1时,表示该扇区中的页已从海量存储器装入磁盘。通常外页表存放于辅存中,当发生缺页中断需要查用时才将它调入主存。从虚页号找外页表可由软件实现。

M	磁盘机号	柱面号	磁头号	扇区号
---	------	-----	-----	-----

图3 外页表项的格式

虚拟存储器工作过程

虚拟存储器工作过程可归纳如下(以页式虚拟存储器为例):

- (1) 根据虚地址中的虚页号查页表。
- (2) 当页表中对应于该虚页的页表项中的装入位为1时,表示该页已在主存中,页表项中的实页号和页内地址拼接为主存实地址。
- (3) 按主存实地址去访问主存(绝大部分主存访问到此为止)。
- (4) 当页表中对应于该虚页的页表项中的装入位为0时,产生缺页中断。

下面转入管态,由操作系统处理缺页中断:

- (5) 根据虚地址查外页表。
- (6) 当外页表中对应于该虚页的外页表项中的装入位为1时,表示该页在辅存中,从外页表项可得

到辅存实地址,按辅存实地址去访问辅存中的这一页。

(7) 查主存页面表,以确定从辅存中调出的页放入主存中的位置。

(8) 如果主存未装满,只需确定1个空的页面即可。

(9) 如果主存已装满,根据替换算法确定替换页面。

(10) 将上述(8)和(9)确定的空页面或替换页面的实存页面号送入通道(或输入输出处理机)。

(11) 在进行页面替换前检查被替换的页在进入主存之后是否已被修改,如果未被修改,通道从辅存中读出要调出的页并写入主存的指定页面。

(12) 如果被替换的页在主存中已被修改,需先将该页写回它在辅存中原来的位置,然后再从辅存中读出要调出的页并写入主存的指定页面。

缺页中断的处理到此结束,下面转回目态,原来被中断的程序继续运行。

(13) 如果在执行完上述(5)以后,在外页表中查出对应于该虚页的外页表项中的装入位为0,则表示该页不在辅存中,这时需要进入另一层中断,将海量存储器中的这一页调入辅存,然后退出这层中断,继续执行(6),将这一页从辅存调入主存。

参考文献

孙强南,孙显东. 计算机系统结构. 北京:科学出版社,1992
(孙强南)

xunjij

虚拟机(virtual machine) 带有解释器(或称解释程序)的抽象机。虚拟机通常用来实现强调移植性的高级语言。高级语言程序先编译为抽象机代码指令序列,然后由**汇编语言**或**C语言**等实现的解释器加以解释执行。

JAVA虚拟机JVM是一个典型的例子。JVM规范定义了一个包括指令系统、寄存器集、堆栈、无用单元堆和存储区的抽象机。JVM的实现可以基于实际的处理器指令,也可以基于微芯片。一旦系统安装了JVM,编译后的JAVA程序,即JAVA字节代码,便可在其上运行。JAVA虚拟机使JAVA编写的程序一次编译到处可用成为可能。

运行JVM字节代码的工作是由解释器完成的。字节代码的解释分为三步:代码装入、代码校验和代码执行。字节代码的执行有两种方式:即时编译方式,解释器先将字节码编译成机器码,然后再执行

该机器码;解释执行方式,解释器通过每次解释并执行一小段代码来完成 Java 字节代码程序的所有功能。

常见的虚拟机还有并行虚拟机(PVM),虚拟机 Lisp 机(VLM)等。

JAVA 虚拟机 JVM 是基于栈的,另外一些虚拟机,如 Perl 6 的虚拟机 Parrot 是基于寄存器的。而并行虚拟机 PVM 则表示一些由网络连接的处理器的集合,其中每个处理器既可以对局部存储器进行存取,也可以对网络上的共享存储器进行存取,它甚至不要求处理器所在的每台机器都运行相同的操作系统。

虚拟机的另外一种含义是物理计算环境的软件模拟,通过系统控制程序为不同的操作系统或程序的执行提供的一个模拟的执行环境,其目的是在单一的机器上支持不同操作系统或一个操作系统的不同版本的同时运行,这方面的例子有 VMWARE 和 IBM 的 VM/ESA 等。

有时,虚拟机也指多用户操作系统为用户提供的共享资源环境,其中每个用户的感受是独享所有计算机资源。(黄林鹏)

xuni xianshi

虚拟现实(virtual reality, VR) 计算机生成的具有临场感的虚拟世界(环境)以及生成虚拟世界所涉及的方法与技术。它也是一种全新的人机交互系统。virtual environment, artificial reality, cyberspace 等都是虚拟现实的同义词。虚拟现实(VR)的特点是:①计算机生成的是一个给人多种感官刺激(理论上应包括视、听、触、嗅、味觉等刺激,目前实际应用的仅为前3种)的虚拟环境;②虚拟世界给人一种身临其境感,又称沉浸感;③用户能以自然方式与虚拟对象进行交互操作。这3点都是VR第一次提出的,有别于其他计算机应用学科的鲜明特点。

1965年美国I. E. Sutherland发表了“The Ultimate Display”一文,提出了一种让人在计算机生成的虚拟世界中具有身临其境感的图形显示技术。这就是他在1968年研制成功的头盔式显示器。这种新型头盔式显示器立即于20世纪70年代在战斗机飞行模拟器等领域获得应用。80年代中期美国国家宇航局(NASA)研制成功用于空间技术的VIVED和VIEW两个VR雏形系统,是VR走向成功的重要一步。随着信息处理技术进步和VR相关硬件价格

下降及VR的成功应用,在20世纪80年代末、90年代初,VR成为信息领域研究、开发和应用的热点。1992年3月美国国家科学基金会召开虚拟现实研讨会,对虚拟现实环境的定义及其研究方向提出了详细建议,更奠定了虚拟现实作为独立研究方向的地位。

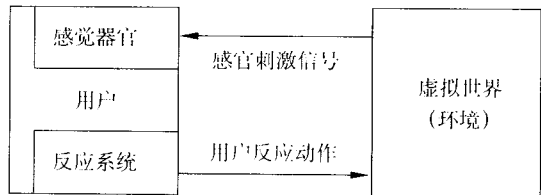


图1 虚拟现实的概念模型

虚拟现实(VR)的概念模型如图1所示。它由计算机创建的虚拟世界和用户两部分组成。VR的核心是强调两者之间的交互操作,即用户感知虚拟世界提供的各种感官刺激(包括光、声、力等)信号并对其作出各种反应动作(如语言与头、手、肢体和身驱动作);虚拟世界检测并辨识用户各种动作,进而对虚拟对象作相应变更。

VR系统的典型配置及部件功能见图2。

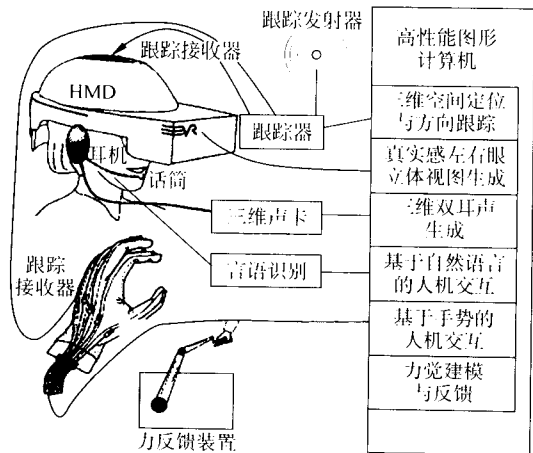


图2 基于头盔显示器的虚拟现实系统硬件配置示意图

虚拟现实的研究内容包括虚拟世界创建和人与虚拟世界之间的自然交互操作两部分。虚拟世界创建部分包括针对人的视觉、听觉和触(力)觉感知通道建立相应的感知模型,以及将各类感知模型转换为刺激信号的装置,如立体显示器、空间声播放和触(力)觉反馈装置等。人机自然交互操作部分包括

基于手势和姿态、基于自然语言理解、三维抓取与指点和三维导航等方法学内容,以及相应的交互设备,如传感手套(衣服)、位置跟踪器、触觉反馈装置和力觉反馈装置等。

虚拟现实是一个多学科交叉的研究领域,集中体现了计算机软硬件、传感器、多通道人机接口、认知和感知等理论和技术的最新成就。VR 也是一个面临众多挑战的研究领域。VR 强调的沉浸感要求人所看到的、听到的和触摸到的计算机生成的虚拟对象都是真实可信的,即强调 VR 系统对人的多感官刺激信号应在物理上和认知上符合人的已有经验。VR 强调的自然方式人机交互操作,目前能提供的方法有限,操作不便、三维定位及定向困难、精度较低等都有待进一步完善。限于技术发展水平,这两个目标至今未能完全实现,从而导致今日沉浸式和非沉浸式 VR 系统并存,自然方式人机交互与传统人机交互共用的现状。

虚拟现实适用于:①未知的宏观和微观世界的探索;②恶劣和危险环境的仿真;③替代制造业的实物样机;④远程协同工作、教育、医疗等。VR 的应用方式可分为真实世界仿真和抽象概念建模两大类。真实世界仿真类应用实例有飞行模拟,战争的战略和战术演练,飞机、汽车等实物样机,手术规划和模拟等。抽象概念建模类应用实例有综合环境模型建立与评估、自然灾害预测预报、新型药物分子结构合成、娱乐游戏,等等。

虚拟现实的发展是使计算机生成的虚拟世界与现实世界相互融合,派生出一个称为增强现实的新方向。增强现实是指用户在看到现实世界的同时,可以看到叠加在其上的计算机生成的虚拟对象和有关信息。用户可以利用虚拟对象和信息来增强对现实世界的理解。目前,增强现实又进一步与随身计算技术相结合形成随身增强现实技术。

参考文献

1. Research Directions in Virtual Environments. Report of an NSF International Workshop. March 23 - 24, 1992, University North Carolina at Chapel Hill. Computer Graphics, 1992, 26(3): 153 ~ 177
2. 汪成为, 高文, 王行仁. 灵境(虚拟现实)技术的理论·实践及应用. 清华大学出版社, 广西科学技术出版社, 1996
3. 石教英主编. 虚拟现实基础及实用算法. 北京: 科学出版社, 2002 (石教英)

xuni xianshi shuchu shebei

虚拟现实输出设备(output devices of virtual reality)

虚拟世界生成的感官刺激信号的输出装置。主要为立体显示装置、空间声播放装置和触(力)觉反馈装置。空间声播放的技术难点在于三维真实感声音生成技术,其播放设备可直接采用大众的多通道音响设备。这里重点介绍立体显示装置和触(力)觉反馈装置。

立体显示装置

立体显示装置是虚拟现实(VR)系统最具特色的硬件组成之一,应具备让用户沉浸在虚拟世界之中的功能。第一个能满足这一要求的显示装置就是 Ivan Sutherland 于 1968 年研制成功的头盔式显示器(HMD)。HMD 只能供个人使用,为了能让几个用户同时沉浸在同一虚拟世界之中进行协同工作,发明了一种称为洞穴式(CAVE)立体显示装置。另外,适用于非沉浸式 VR 的立体显示装置有响应式工作台(RWB)和墙式(半圆柱式)宽屏显示装置两种。

(1) HMD 是一种戴在头部的用来显示 VR 创建的虚拟环境的立体显示器。HMD 分遮挡式和透视式两种,使用遮挡式 HMD 时,用户双眼只能分别看到显示在眼前液晶显示屏上的虚拟环境的左右眼视图,真实世界的实际场景被 HMD 遮挡,完全不能看到,从而给人一种沉浸在计算机生成的虚拟环境之中的感觉。透视式 HMD 是增强现实的显示设备。在 HMD 中显示屏离人眼很近,光学系统应能使人眼聚焦在如此近的距离又不易疲劳,能放大图像又要提供尽可能宽的视野。设计这种光学系统成为 HMD 的关键技术。

(2) CAVE 显示装置是一种由投影显示屏围成的一个像小房子那样的立体显示装置,立体图像由投影仪直接投在显示屏上。这种小房子的形状通常为立方体,边长大于 2 m,像一个洞穴,故称为 CAVE 立体显示装置。在 CAVE 内可容纳 3 ~ 4 人。CAVE 显示装置有 5 个投影屏和 4 个投影屏两种,后者由左、中、右 3 面及地板构成投影屏。用户位于立方体之中时,只能看到背投式显示屏上显示的计算机生成的立体图像,从而产生身临其境的视觉感受。

(3) 响应式工作台(RWB)是一种非沉浸式,支持多用户协同工作的立体显示装置,是德国信息科学研究中心(GMD)的科学家于 1993 年发明。RWB 立体显示装置是一台式装置,桌面兼作显示屏,其尺

寸约为 $1.8\text{ m} \times 1.2\text{ m}$ 。显示屏下安装一块大的反射镜,安装在后部的投影仪将立体图像投射到反射镜面上,由反射镜将图像反射到显示屏上。显示在屏上的可以是虚拟对象,也可以是各种控制菜单。用户佩戴立体眼镜可以看到跃现在屏面之上的立体感很强的虚拟对象。由于显示屏尺寸较大,因此允许许多用户观察并协同操作虚拟对象。1998 年 GMD 科学家又发明了双屏 RWB 立体显示装置。这是一个“L”形的由两个正交显示屏组成的立体显示装置,立体效果优于单屏 RWB。

(4) 墙式(半圆柱式)宽屏立体显示装置顾名思义是一种类似于宽银幕电影那样的立体显示装置,图像由多台投影仪投射图像拼接而成。根据投影仪和配套立体眼镜种类的不同,墙式(半圆柱式)立体显示装置又可分为主动系统和被动系统两类。主动系统采用有源立体眼镜,被动系统采用无源立体眼镜。此类立体显示装置的最大优点是允许多达数十人共享虚拟环境。

触觉与力觉反馈装置

触觉反馈装置目前局限为手指触觉,且仅能提供最基本的“触到了”的感觉,无法提供材质、纹理、温度等感觉。已实用的触觉反馈装置有充气式和振动式两类:①充气式触觉反馈装置的工作原理是在手套中配置一些微小气泡,这些气泡可以按需要充气 and 排气。充气时气泡膨胀而压迫刺激皮肤达到“触到了”的感觉。小气泡安放在手套的手指和手掌部位。②振动式触觉反馈装置的工作原理有利用音圈产生振动和利用形状记忆合金微型触头的伸缩达到触觉反馈。后者通常组成触头阵列,因此还能提供触觉的位置感。

力觉反馈装置已成为产品的有机机械臂式和操纵杆式两种。①机械臂式力觉反馈装置由一个有 4 个自由度的机械臂和操纵者手持的带 6 个自由度的力觉反馈传感器组成。计算得到的力经过数模转换去控制机械臂关节上的小马达,产生反馈力,传感器测量机械臂传递给操作者的力和力矩。②操纵杆式力觉反馈装置外形与游戏杆相似,操纵杆架在两个马达带动的可调节轴承上,操纵杆在每一轴上(x 和 y)能产生 75g 力,动作范围约为 55mm。改进型操纵杆式力反馈可产生三维的力。

参考文献

石教英主编. 虚拟现实基础及实用算法. 北京:科学出版社,2002

(石教英)

xuni xianshi shuru shebei

虚拟现实输入设备(input devices of virtual reality)

检测并跟踪虚拟现实的用户头、手、躯体等部位的位置、朝向和姿态的设备。虚拟现实输入设备包括位置跟踪器和传感手套(衣服)两大类。虚拟现实输入设备将用户的位置、朝向和姿态等信息输入到 VR 系统,确定用户的视点位置和视线方向,用于选择虚拟场景的可见部分及识别用户的手势和体势用作人机交互信息。

位置跟踪器 为了确定和跟踪物体在三维空间中的位置和朝向,位置跟踪器应能检测 x, y, z 坐标位置以及水平角、俯仰角和偏转角,即具有检测 6 个自由度数据的功能。常用位置跟踪技术有 5 类:磁跟踪技术、光学跟踪技术、声学跟踪技术、机械跟踪技术和惯性跟踪技术。今简要介绍在 VR 系统中常用的前 4 类位置跟踪技术。

(1) 磁跟踪技术采用磁场进行位置和方位跟踪,由一控制部件、几个发射器和接收器组成。根据发射磁场的不同,又可分为交流发射器型和直流发射器型两种。交流发射器和接收器各由 3 个互相垂直的线圈组成。发射器产生的 3 个相互垂直磁场分量在空间传播,在接收器的线圈上产生感应电流。感应电流的强度与其距发射器的距离有关。通过电磁学计算,可以从 9 个感应电流求得发射器和接收器之间的距离和角度。直流发射器结构和交流发射器类似,两者不同之处在于它发射的是一串直流脉冲磁场,形成一个开关式的磁场向外传播。直流发射器能避免金属物体的干扰。

(2) 光学跟踪技术有两种实现方法,即标志法和激光测距法。标志法可以将一个或几个发射器(即光源)固定在被跟踪的运动物体上,一些固定安置在空间中的传感器从外面“看”着发射器运动,计算被跟踪物体的运动情况。这种标志系统称为“自外而内”结构。另一种结构称为“自内而外”,就是将光源和传感器的位置互换。标志法是目前使用最多的光学跟踪技术。激光测距法通过比较从激光源发射的衍射光栅信号与经物体表面反射的二维衍射信号差来测量光源与物体间的距离。方位由激光发射方向确定。激光测距法的优点是无需在被跟踪对象上安装发射器接收器。

(3) 声学跟踪技术采用超声波,通过飞行时间测量法和相位干涉测量法来测量声源和物体之间的距离和方位。

(4) 机械跟踪技术的原理是通过 6 自由度机械

臂上的基准点与被测物体相接触来检测其位置和方位。

传感手套 传感手套是一种带传感器的手套,用于基于手势的人机交互操作,是VR的创新技术之一。这里,一方面需要赋予各种手势确切的交互操作的含义,另一方面传感手套能把手指和手掌伸屈时的各种手势转换为数字信号送计算机识别、执行。在实际应用中传感手套还要配备位置跟踪器来检测手的空间位置和朝向。现在已有多种传感手套问世,它们之间的区别在于采用的传感器不同。

(1) 数据手套是VPL公司于1987年推出的一种传感手套。手套是用轻质的富有弹性的材料制造,采用光纤作为传感器,每个手指的每个关节上装有一个光纤回路,用于测量手指每个关节的弯曲角度。光纤回路的一端为发光二极管,作为光源。另一端为光敏二极管,检测经光纤回路返回的光强。当光纤伸直时,光的传输无衰减。在手指关节处,光纤壁做过专门处理,使得手指弯曲时光会逸出光纤,其逸出量与关节弯曲程度成比例。因此,经过校准,测量返回光强就可测出手指关节的弯曲角度。

(2) 赛伯手套的外观类似于数据手套,但在每一关节处织有由两片应变电阻片叠成的传感器。当手指弯曲时,下面一片受到挤压,上面一片受到拉伸,从而引起电阻变化。测量电阻变化求得手指关节的弯曲角度。

(3) DHM手套是一种金属结构的传感手套。DHM手套安装在手背上,每个手指关节处安装一只基于霍尔效应的位置传感器。DHM手套安装和脱卸过程较麻烦,但采样速度快,精度高,分辨率高。

参考文献

石教英主编. 虚拟现实基础及实用算法. 北京: 科学出版社, 2002 (石教英)

xuni zhongduan

虚拟终端 (virtual terminal, VT) 将计算机网络中各种类型的实终端的功能一般化、标准化后得到的一种终端模型。网络中的应用程序和实终端经过映像后,变换成标准的虚拟终端进行通信。

在计算机网络环境中,任一实终端要访问网络中的应用程序,都必须为该应用程序所接受,即必须符合应用程序与终端的接口要求,如不符合,则需要有一个实体来进行映像。若一个计算机网络中有 m 种终端和 n 种应用程序,则必须有 $m \times n$ 个映像实

体,这无疑会大大增加每个计算机系统的开销。如果在网络中建立一个标准的、规范化的抽象终端模型,任何实终端或应用程序均与其建立标准接口,则就将 $m \times n$ 的问题简化为 $m + n$ 的问题,且在扩充新类型的终端或应用程序时,只要增加一个新的映像实体即可。

虚拟终端模型如图1所示,它由下列3个部分组成:

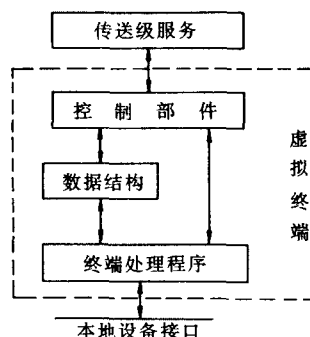


图1 虚拟终端模型

(1) 控制部件 实现虚拟终端协议(VTP),并实现与传送级服务接口;

(2) 数据结构 包括正文数据及其属性,并记录用户与应用程序之间通信的当前状态。不同类型的虚拟终端所需的数据结构不同;

(3) 终端处理程序 提供虚拟终端与本地设备(如输入、输出等设备)之间的接口。

由于计算机终端种类繁多,各种实际应用中对终端的要求也不同,要想将所有终端的众多功能归结在一起,抽象化、一般化为一个虚拟终端模型是很困难的。即便如此,也不利于终端功能的扩充。一个有效的方法是按类建立虚拟终端模型及其协议。国际标准化组织将虚拟终端分成5类:

(1) 基本类 处理由字符元素或镶嵌元素组成的一维、二维或三维数组,对应的实终端有电传打字机、页式显示终端等;

(2) 表格类 处理对象与基本类相同,另外还具有对输入、输出进行表格格式控制处理的能力,对应的实终端有处理各类业务(如航运、银行等)的窗口服务终端;

(3) 图像类 处理由像元元素组成的二维或三维数组,对应的实终端是图像处理终端;

(4) 图形类 处理由几何图形元素(如点、线、二次曲线段等)组成的多层结构,对应的典型实终

端是图形终端;

(5) 混合类 处理由字符元素、镶嵌元素、像元元素和几何图形元素组成的多层结构,它是以上4类的混合,对应的典型实终端是高级工作站。

为了提供网络环境下使用分时系统的功能,一般要在提供服务的远地主机处引入虚拟终端,并在其分时系统中配置相应的虚拟终端控制程序。这种虚拟终端实际上是由虚拟终端协议服务进程实现的,它与实际物理终端无关,完全是按虚拟终端协议的规定,进行公共处理而产生所需的控制信息的数据代码。另一方面,为了便于用户通过自己的物理终端或用户程序去使用远地分时系统,则启动虚拟终端协议使用进程,按照虚拟终端协议的规定,把本地系统所支持的终端映象成标准虚拟终端。其功能包括:与远地主机之间建立或拆除一组连接、传输控制信息和数据信息、产生中断信号等。这样,用户就能通过自己的终端访问远地分时系统,传送数据和文件内容,控制作业运行,调用各种服务子系统。ARPA网的虚拟终端协议称为Telnet,是针对基本类终端设计的。

参考文献

1. 曹东启等. 计算机网络软件基础. 北京: 人民邮电出版社, 1982
2. 胡道元. 计算机局域网. 北京: 清华大学出版社, 1990 (李学农)

xuni zhuanwang

虚拟专网(virtual private network, VPN)

将物理分布在不同地点的网络通过公用骨干网,尤其是Internet,连接而成的逻辑上的专用网络。为了保障信息的安全,虚拟专网(VPN)技术采用了鉴别、访问控制以及保证数据保密和数据完整性等措施,以防止信息被泄露、篡改和复制。

基于Internet的VPN具有节省费用、运行灵活、易于扩展、易于管理,且能保护信息在Internet上安全传输等优点。企业可以利用VPN技术和Internet构建安全的企业内联网和外联网。

虚拟专网(VPN)的模式有两种:直接模式和隧道模式。直接模式的VPN使用网际协议(IP)和编址来建立对VPN上传输的数据的直接控制。采用基于用户身份的鉴别,而不是基于IP地址的鉴别。隧道模式使用IP帧作为隧道发送分组。大多数VPN都运行在IP骨干网上,数据加密通常有三种方法:具有加密功能的防火墙、带有加密功能的路由

器、单独的加密设备。

通常的包过滤防火墙只能控制来自或去往某个站点的IP数据包的通过,可以拒绝来自某个外部站点的IP数据包访问内部某些站点,也可以拒绝某个内部站点对某些外部网站的访问。但包过滤防火墙不能保证从内部网络出去的数据包不被截获,也不能保证进入内部网络的数据包未经篡改。只有对IP数据包实施了加密和鉴别后,才能保证在外部网络传输的数据包的保密性和完整性。

一种对IP数据包进行加密和鉴别的技术是网际协议安全性协议(Ipsec)。这是一种由Internet工程任务部(IETF)制定的用于虚拟专网(VPN)的协议。为了进行加密和鉴别,还需要有密钥的管理和交换的功能。通信双方如果要用Ipsec建立一条安全的传输通路,需要事先协商好要采用的安全策略,即使用的加密算法、密钥、密钥生存期等。当双方协商好所使用的安全策略后,就表示双方已建立了一个安全关联(SA),已确定了Ipsec要执行的处理。Ipsec协议分三个部分:封装安全负载协议(ESP)、鉴别报头(AH)、Internet密钥交换(IKE)。封装安全负载协议(ESP)主要用来处理对IP数据包的加密,并对鉴别提供某种程度的支持。鉴别报头(AH)只涉及鉴别,不涉及加密。除了对IP的有效负载进行鉴别外,AH还可对IP报头实施鉴别。Internet密钥交换(IKE)协议主要是对密钥交换进行管理。

Ipsec是用于VPN的一项新技术,一些主要的网络厂家已推出相应的产品,但这些产品在兼容性和网络性能方面仍有待改进。

参考文献

- 胡道元主编. 网络设计师教程. 北京: 清华大学出版社, 2001 (胡道元)

xuqiu dingyi yuyan

需求定义语言(requirements definition language)

用于书写软件需求定义的语言。软件需求包括功能需求和非功能需求两个方面。功能需求从用户角度明确了软件系统必须具有的功能行为,它是整个软件需求的核心所在。在功能需求的基础上,非功能需求对软件需求作进一步的刻画,它包括功能限制、设计限制、环境描述、数据与通信规程和项目管理等。软件需求定义主要面向用户,采用基于现实世界的描述模型,以便于用户理解。

需求定义语言的研究受计算机应用发展的推

动,其发展大致可分为三个阶段。

从20世纪60年代末期到70年代初期为第一阶段。在计算机发展早期,由于问题规模较小,需求定义语言研究的重要性并未引起足够重视,人们常采用自然语言来书写一些简单的需求。自然语言对于规模较小的应用还能够应付,但对于大型软件系统,其内在的非形式性导致需求定义中经常出现错误,并且由于由机器自动处理自然语言的非形式性异常困难而使得纠正需求定义中的错误不仅代价高而且费时。随着计算机应用规模的不断扩大,特别是自1968年在大西洋公约学术会议上提出软件工程以来,人们逐渐认识到需求定义语言的重要性,开始研究各种类型的需求定义语言。

从70年代初期到80年代中期为第二阶段。由于认识到非形式自然语言给需求定义带来的种种不足,人们以**软件方法学**为基础,开始研究需求定义语言的形式化问题。提出了诸如基于自顶向下途径的SA,基于自底向上途径的问题陈述语言PSL,以及基于面向对象思想的需求定义语言RML等,这类语言的重要特征是较自然语言有比较精确的语法和语义定义,从而便于分析其各种性质,利于用计算机提供自动化的支持。

80年代中期迄今为第三阶段。随着软件系统复杂性的提高和规模的增大,需求定义愈加困难和耗时;传统的需求定义语言由于采用的模型和问题领域差距较大,从而使得需求定义的易理解性和易维护性较差。而面向对象模型由于和问题结构之间有良好的对应关系而较好地满足了需求定义的需要。因此,面向对象需求模型的研究成为热门课题并展示出良好的前景,代表性的工作有J. Rumbaugh等提出的模型以及D. W. Embley等提出的模型。这些模型以对象及其相互间的关系为核心,以图形化表示机制为手段来刻画系统,不仅能反映系统的静态结构关系,而且能反映系统的动态变化行为,为解决传统功能分解模型中存在的功能本身的易变性、分解结构的随意性以及功能结构与问题结构常难对应等问题提供了有效途径。

按照形式化的程度,需求定义语言可分为非形式需求定义语言,半形式需求定义语言,以及形式需求定义语言三类。

非形式需求定义语言是指未作任何限制的自然语言。一般说来,非形式需求定义语言具有易理解、易使用的特点,易于为一般用户所接受。但是由于自然语言的非形式性而使得需求定义中常出现错

误,并且难以用计算机系统提供自动化的支持。半形式需求定义语言是指在宏观上对语言的语法和语义有较精确的描述,而在某些局部方面则允许使用非形式的自然语言。这类语言既便于用户的表达和理解相应的需求定义,又可在某种程度上用机器对相应的需求定义进行管理和进行部分的正确性检查。形式需求定义语言是指其语法和语义均精确定义的语言。一般说来,形式化需求定义语言具有良好的数学基础,易于分析需求定义的各种性质。然而,形式需求定义语言往往要求用户具有较高的数学修养,且相对说来,所书写的需求定义较难理解。目前,形式需求定义语言的研究正在进行之中,并且在非功能需求的形式化方面进展缓慢。

需求定义语言的研究主要涉及基本模型、**语言结构和语用分析**等。

需求定义语言的基本模型是相应软件开发风范在语言中的具体体现。它是表达软件需求的基础,决定了参加需求分析的各类人员的思维方式。两类重要的模型分别是功能分解模型和面向对象模型。在功能分解风范下,语言的用户通常将待解的问题抽象成需要满足的功能,通过功能分解的方法来表述系统各个部分之间的关系,以此为架构来刻画用户对系统的需求。而在面向对象风范下,语言的使用者可以比较直接地刻画现实世界模型,并在此基础上描述对所需软件的需求。从而使得相应需求定义易于理解、易于修改和易于复用。不同的基本模型往往适合于不同的应用领域,并对语言结构的选取以及相应的支撑方法产生重要影响。

语言结构提供了用于刻画系统需求的具体手段,主要包括基本模型的描述、数据描述、控制描述、抽象机制以及项目相关信息描述等,当然,不同语言的侧重点有所不同。在功能分解模型的描述方面,SA采用自顶向下、逐层分解的功能模型,为了支持这一模型,SA提供了分层的数据流图。而PSL则采用了自底向上的途径,它提供了各种实体类型及其相互间的关系用以分别描述各个需求,然后形成所需的需求定义。当然,并不是所有语言均基于单一模型的单一方法,例如,RSL试图将自顶向下和自底向上的途径加以结合以适应实时系统描述的需要,它提供了R网等设施。在面向对象模型支持方面,通常有两种途径,其一是设计新的需求模型和语言来支持面向对象方法,例如,D. W. Embley等提出的面向对象需求模型,提供了各种图形化表示机制来刻画对象及其相互间的关系。其二是用已有

语言来刻画面向对象模型,例如,ROOA 方法采用形式规约语言 LOTOS 中的抽象数据类型和进程定义来描述面向对象需求定义,并对继承机制提供支持。在数据描述方面,PSL 提供了 ENTITY, CONSISTS OF, DERIVEDBY 等实体或关系来描述数据对象的名、数据结构和数据流程。在控制描述方面,RSL 提供了各种类似于程序设计语言中使用的控制结构。尽管此类结构对于通信系统和实时系统较为有用,但可能导致与实现有关的问题。在抽象机制方面,RML 语言在面向对象的架构下提供了聚合、分类和泛化三类抽象机制,它们可一致地用于语言中的三类规约单位:对象、活动、断言。在项目相关信息描述方面,各类半形式化的语言均提供了描述这些信息的手段。例如,PSL 可描述文档信息,这些信息通常是借助于自然语言来刻画的。

语用分析主要讨论需求定义语言的适用领域、可扩展性等性质,以及相应的方法与工具支持。例如,PSL 主要适合于商业应用,RSL 主要用于实时系统;而有些语言如 SA 则是通用语言;可扩展性指是否可在原有语言结构基础上从语法和语义角度定义新的语言结构;在此意义下,PSL 和 RSL 本质上是可扩展的。为了充分发挥需求定义语言的作用,必须研究与之相应的方法和工具。方法应给出获取需求的原理和步骤以及如何从需求开发相应的程序。而工具支撑则对需求的分析提供自动支持。在某种意义上,几乎所有语言均对如何形成需求提出建议和提示,但真正的方法却不多见。就工具支撑而言,SA 是为手工使用而设计的,而 PSL, RSL 则一开始就将工具支撑和语言设计集成在一起。

概括起来,需求定义语言的研究已取得较大进展,已有的各种类型的需求定义语言已逐步应用于软件工程实践并取得良好效果。近年来,国际上又兴起需求工程的研究热潮,1993 年召开了第一届需求工程国际研讨会,1994 年召开了第一届需求工程国际会议,国际信息处理协会 IFIP 成立了需求工程工作组,即 IFIP WG 2.9。需求定义语言是需求工程的核心内容之一,因此,可以预言,需求工程的研究必将促进需求定义语言的进一步发展。

参考文献

1. McDermid J A (ed). Software Engineer's Reference Book. Butterworth-Heinemann Ltd., 1991
2. Partsch H A. Specification and Transformation of Programs. Springer-Verlag, 1990 (吕建)

xuqiu fenxi gongju

需求分析工具 (requirements analysis tool)

软件开发过程中用于需求分析活动的软件工具。它辅助系统分析员从需求定义出发,生成完整的、清晰的、一致的功能需求规约(参见软件需求定义与功能规约)。

由于需求分析活动本身的复杂性,目前尚无支持全部需求分析活动的工具出现,现有的需求分析工具只能支持需求分析中的部分活动。

按描述需求定义的方法可将需求分析工具分为基于自然语言或图形表示的工具,和基于形式化需求定义语言的工具。基于自然语言或图形描述的工具采用分析与抽象等基本手段对用户问题逐步求精,并在检测机制的辅助下,发现其中可能存在的问题,通过对问题描述的修改,逐步形成能正确反映用户需求的功能规约。这些工具给分析员的帮助体现在:提高需求文档的质量,降低功能规约的维护费用。常用的图形描述方法有数据流图、实体关系图、状态迁移图等。例如结构化分析工具通常采用数据流图的形式描述用户需求,在图形编辑程序的支持下实现数据流图的逐步求精及抽象,降低需求分析的复杂度。它应该提供相应的检测机制来保证文档的一致性、完备性,并采用数据词典统一存储和管理数据流图中的各类数据,以及关于功能的描述。采用该工具后可得到一套分层的数据流图和一本数据词典。目前,基于图形表示的需求分析工具已有较多产品。

基于形式化需求定义语言的工具大多以基于知识的需求智能助手的形式出现,并把人工智能技术运用于软件工程。这种工具通常具有一个知识库和一个推理机制,接收用形式化语言书写的功能描述。知识库中存放需求分析所需的公共知识,以及特定的应用领域知识。这些知识能用来理解需求定义中的省略写法,能部分消除不完整性和歧义性。推理机制能容忍需求定义的无序性,部分解决描述中的不一致性。运用知识库中的知识,通过推理,发现需求定义中的矛盾和不足,经补充、更新知识库中的知识和规则,以及与分析员的不断交互,得到完整的功能规约。形式化需求分析工具的研究和开发仍处于起步阶段,目前的研究成果大多是实验性系统,较成功的工具产品还不多见。

此外,可执行规约语言以及原型技术为需求分析工具提供了另一条实现途径。这些工具通过运行可执行规约或原型将有关的结果显示给用户和分析

员,以便他们进行需求确认。

参考文献

1. Davis A M. Software Requirements: Objects, Functions and States. Englewood Cliffs: Prentice-Hall Inc., 1993
2. Gane C. Computer-Aided Software Engineering: The Methodologies, the Products, and the Future. Englewood-Cliffs: Prentice-Hall Inc., 1990
3. Firesmith D G. Object-Oriented Requirements Analysis and Logical Design. New York: John Wiley&Sons Inc., 1993 (钱乐秋 赵文耘)

xuqiu gongcheng

需求工程 (requirements engineering) 实施需求分析的工程。需求分析包括系统需求分析、硬件需求分析、软件需求分析等。它有两层含义。一层含义指从客户或用户提出的不完备、有歧义的需求完备化、一致化,形成可能用半形式化或形式化语言表述的需求定义(亦称需求规约)。亦即,需求分析是收集、澄清客户或用户提出之需求、得出需求定义的过程。另一层含义是,不仅指从客户或用户所提需求出发、得出需求定义,而且还包括从需求定义转换到相应功能规约的过程。

需求分析的基本活动是:

- (1) 需求收集 收集并澄清清为客户或用户所需的必要需求信息;
- (2) 规约形成 将所收集的需求信息陈述成完整文档,一般称作需求定义或需求规约,或再将后者转换成相应的功能规约;
- (3) 质量验证 保证需求定义或需求定义与功能规约的一致性、完备性;
- (4) 规约确认 保证需求定义或需求定义与功能规约能精确描述或反映客户或用户所需的需求。

参考文献

1. Zave P. Classification of Research Efforts in Requirements Engineering. ACM Computing Surveys, 1997, 29(4): 315 ~ 321
2. 徐家福,吕建. 软件语言及其实现. 北京: 科学出版社, 2000 (金芝)

xushu

序数 (ordinal number) 良序集(参见关系)的序型。若令每个线性序集 M 都对应于惟一的一个

符号 σ_M ,使得对任意两个线性序集 M_1 和 M_2 皆有 $\sigma_{M_1} = \sigma_{M_2}$ 当且仅当有从 M_1 到 M_2 的序同构,则称 σ_M 为 M 的序型,并常用 \bar{M} 表示之。

$\langle \mathbf{N}, \leq \rangle$ 的序型用 ω 表示。

有穷良序集的序型称为有穷序数,全部有穷序数显然为 $0, 1, 2, \dots$ 。无穷良序集的序型称为超穷序数。 ω 就是一个超穷序数。

序型加法 设 σ 和 τ 为两个序型,所对应的线性序集分别为 $\langle A_1, R_1 \rangle$ 和 $\langle A_2, R_2 \rangle$,不妨假定 $A_1 \cap A_2 = \emptyset$ 。若令 $R = R_1 \cup R_2 \cup (A_1 \times A_2)$,则 R 为 $A_1 \cup A_2$ 上的一个线性序。通常称线性序集 $\langle A_1 \cup A_2, R \rangle$ 的序型为 σ 与 τ 的和,记为 $\sigma + \tau$ 。显然此定义与 $\langle A_1, R_1 \rangle$ 和 $\langle A_2, R_2 \rangle$ 的具体取法无关。

因为 $1 + \omega = \omega$ 且 $\omega + 1 \neq \omega$,所以序型加法不可交换,但可结合,即对任意序型 σ, τ 和 λ 恒有

$$(\sigma + \tau) + \lambda = \sigma + (\tau + \lambda)$$

故而此和可记为 $\sigma + \tau + \lambda$ 。

序型乘法 设 σ 和 τ 为两个序型,所对应的线性序集分别为 $\langle A_1, R_1 \rangle$ 和 $\langle A_2, R_2 \rangle$,不妨假定 $A_1 \cap A_2 = \emptyset$ 。若在 $A_1 \times A_2$ 上定义二元关系 R 如下:

$$\langle \langle x_1, y_1 \rangle, \langle x_2, y_2 \rangle \rangle \in R \quad \text{当且仅当}$$

$$\langle y_1, y_2 \rangle \in R_2 \text{ 或 } y_1 = y_2 \text{ 且 } \langle x_1, x_2 \rangle \in R_1$$

则 R 显然为 $A_1 \times A_2$ 上的一个线性序。称线性序集 $\langle A_1 \times A_2, R \rangle$ 的序型为 σ 与 τ 的积,记为 $\sigma \cdot \tau$ 。显然此定义与 $\langle A_1, R_1 \rangle$ 和 $\langle A_2, R_2 \rangle$ 的具体取法无关。

因为 $2\omega = \omega$ 且 $\omega 2 = \omega + \omega \neq \omega$,所以序型乘法不可交换。但序型乘法是可结合的,即对任意三个序型 σ, τ 和 λ 恒有

$$(\sigma \tau) \lambda = \sigma (\tau \lambda)$$

故而此乘积可记为 $\sigma \tau \lambda$ 。

序型乘法满足第一分配律,即对任意三个序型 σ, τ 和 λ 恒有

$$\sigma (\tau + \lambda) = \sigma \tau + \sigma \lambda$$

因为 $2(\omega + 1) \neq (\omega + 1)2$,所以序型乘法不满足第二分配律。

设 σ 为任意序数。若有序数 τ 使 $\sigma = \tau + 1$,则称 σ 为后继序数或非极限序数,否则称 σ 为极限序数。有穷序数和 $\omega + 1$ 都是后继序数, ω 与 $\omega + \omega$ 都是极限序数。

序数的幂 设序数 $\sigma \neq 0$,则 σ 的幂定义如下:

$$(1) \sigma^0 = 1;$$

$$(2) \text{ 若 } \tau \text{ 为后继序数,即有序数 } \lambda \text{ 使 } \tau = \lambda + 1, \text{ 则 } \sigma^\tau = \sigma^\lambda \cdot \sigma;$$

(3) 若 τ 为极限序数, 则 $\sigma^\tau = \lim_{\xi < \tau} \sigma^\xi$ 。

(王兵山)

xuexi jisuan lilun

学习计算理论 (learning computation theory) 采用数学方法, 研究学习算法的计算复杂性和所需信息量的大小, 分析算法所需的时间和空间资源, 判定学习对象的可学习性等所形成的理论。

学习计算理论的研究始于 20 世纪 60 年代。E. M. Gold 于 1967 年在形式语言研究方面提出极限辨识理论, 给出了什么是正确的辨识(学习)的形式定义。D. Angluin 探讨了新的学习对象, 提出了模式语言的学习, 通过对学习对象的限制, 获得了某些有用的结果。1984 年, L. G. Valiant 提出了有名的概率近似正确 PAC 模型。PAC 模型较好地反映了机器学习与人类学习的某些特点。另外, A. Blumer 等人将 Vapnik-Chervonenkis 维数(简称 VC 维数)和可学习性以及学习所需样本数联系起来, 获得了不少有用的研究成果, 如在神经网络泛化理论上的进展。1987 年, 从学习协议的角度探讨了一种新的学习——EXACT 学习, 它允许学习者向谕示询问, 基本的询问有所属性和等价性两种。它能解决基本 PAC 模型下不能学习的某些概念类。

在学习计算模型中, 传统的代表模型是 E. M. Gold 的极限辨识模型, 到现在仍有不少的工作与之相关。相对来讲, L. G. Valiant 的 PAC 模型由于更接近于实际和考虑到可行性问题, 现在则更为流行。下面就这两种模型及统计学习理论给出较为详细的描述。

(1) 极限辨识模型 源于形式语言的文法推断。它把归纳推理看作是一种无限的过程, 衡量辨识或学习的成功与否依赖于归纳的最终或极限行为, 即是否在某点收敛。形式地讲, 设 M 是一个试图正确描述某个未知概念 c 的学习算法, M 的输入是概念 c 的实例, M 根据这些实例输出它对概念 c 的猜测 G , 随着实例越来越多或实例数任意地增大, 猜测 G 便形成一个猜测序列 G_1, G_2, G_3, \dots 。如果存在某个数 m , G_m 是 c 的正确描述, 即 $G_m = G_{m+1} = G_{m+2} = \dots$, 那么称 M 极限辨识概念 c 。 M 也可以看作是对未知概念 c 不停地进行学习, 不停地修改它关于 c 的猜测。如果有限次后, M 停止修改它的猜测(虽然实例还在不停地输入), 这个最后的猜测就

是 c 的正确描述, 这时 M 就是极限辨识了概念 c 。注意, M 自己并不能知道它是否收敛到一个正确的假设, 因为新的实例与当前的猜测是否冲突并不知道。如果 M 对概念 c 所属的概念类 C 的每一个概念都能进行极限学习, 则 M 能对 C 进行极限学习。对于概念类 C 来说, 若存在这样的算法 M , 则称 C 为可极限学习。

(2) PAC 学习模型 和以往的模型相比, PAC 模型有两个特点, 一是允许学习者有时失败, 二是允许学习者在学习成功时可以是近似正确的。在给定的误差参数 ϵ 和可靠性参数 δ 下, 学习成功意味着学习产生的假设和目标之间的误差不超过 ϵ 。当误差超过了 ϵ 时, 学习则失败了, 但要求这种失败的概率小于 δ , 即成功的概率不低于 $1 - \delta$ 。这就是 PAC 模型关于可学习性的定义, 它也称之为可 PAC 学习。这种模型所要研究的主要是样本复杂性和时间复杂性, 即为了学习某个概念类, 到底需要多大的样本, 能否在 $1/\epsilon, 1/\delta$ 的多项式时间内进行学习。形式定义则为: 对任意的概率分布 P 和任意的 $0 < \epsilon, \delta \leq 1$, 对所有的概念 $c \in C$, 如果存在算法 A , 能在 $1/\epsilon, 1/\delta$ 和表示概念 c 的复杂度的某种编码长 n 的多项式时间内输出假设 c' , 使得

$$\text{Probability}\left(\sum_{c(x) \Delta c'(x)} P(x) \leq \epsilon\right) \geq 1 - \delta$$

则称概念类 C 是多项式时间可学习的。上式中的 $c(x) \Delta c'(x) = (c(x) - c'(x)) \cup (c'(x) - c(x))$ 。

由于 PAC 模型较其他学习模型相对地更为有效, 因此目前学习计算理论方面所作的工作大多是和 PAC 模型有关的。但是在 PAC 模型下, 仍有许多有趣的学习问题是不可学习的, 下面是几个较有代表性的情形。例如在基本 PAC 模型下, 确定型有限自动机是不能多项式学习的, 但若允许学习者能够进行提问, 则是可多项式学习的。又如, 基本 PAC 模型没有考虑噪声问题, 而实际中噪声是必不可少的。

(3) 统计学习理论 统计学习理论是利用统计学原理研究机器学习的规律、原则和方法的理论, 通常特指 Vapnik 等人开创的研究有限样本下机器学习问题的理论。机器学习是指用数学方法从已有观测数据中寻找用于预测未知数据的规律, 模式识别是最典型的机器学习问题, 它研究如何根据已知类别的样本数据设计分类器, 使之能对未知样本给出分类决策, 目标是这种决策带来的风险最小。统计学习理论研究起源于 20 世纪六七十年代, 到 90 年

代趋于成熟,其理论和应用在最近十年内得到了飞速的发展。它建立了一套有限样本下统计学习的理论体系,系统研究基于有限样本定义的经验风险与理论上的期望风险之间的关系;在此基础上研究机器学习方法对未来样本正确预测的能力(推广性),得到推广性与样本数目及学习机器复杂性之间的关系;提出了小样本情况下机器学习方法应遵循的准则,即结构风险最小化原则,指出学习机器的复杂性应与样本数目相适应。其中一个核心概念是 VC 维,它是由 Vapnik 和 Chervonenkis 提出的描述学习机器复杂性的一个量度。在这些理论和原则基础上,统计学习理论在 20 世纪 90 年代发展出一种有效的通用机器学习方法,即支持向量机(SVM)方法,它通过兼顾最小化分类错误和最大化与推广性密切相关的分类间隔,达到在有限样本下较好的推广性,并引入核函数内积巧妙地解决了非线性问题。统计学习理论的主要内容也包括函数回归、密度估计这两类更一般的机器学习问题。统计学习理论属于统计学与计算机科学的交叉学科。

学习计算理论的大部分工作目前还是以概念学习为主,尤其是实例学习。而在机器学习的其他领域所作的工作较少,人们也期待着学习计算理论能在这些领域获得进展。

参考文献

1. Natarajan B K. Machine Learning: A Theoretical Approach. San Mateo: Morgan Kaufmann, 1991
2. Shi Zhongzhi. Principles of Machine Learning. Beijing: International Academic Publishers, 1992

(史忠植 王军 张学工)

xunzhi fangshi

寻址方式 (addressing mode) 生成计算机指令中实际使用的有效地址的方法。

不同的计算机有多种寻址方式,下面介绍广泛采用的几种,并以一地址指令为例加以说明。

(1) 直接寻址

指令的地址码部分给出操作数在存储器中的地址,如图 1 所示,图中的寻址方式由操作码 OP 约定, A 为地址。

(2) 寄存器寻址

中央处理器中一般设置有一定数量的通用寄存器,用以存放操作数、操作数地址或运算的中间结果。指令的地址码 A 为通用寄存器地址,操作数在

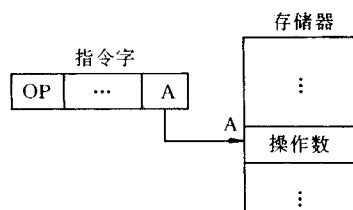
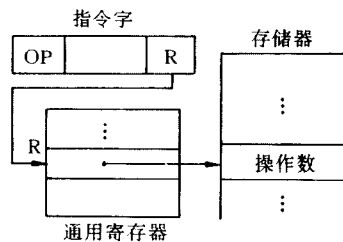


图 1 直接寻址

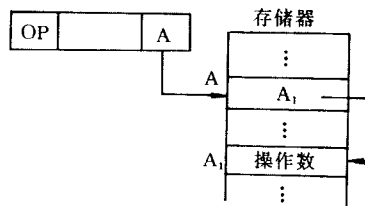
该寄存器中。这种寻址方式也可看作是一种直接寻址。

(3) 间接寻址

根据指令的地址码从存储器或寄存器中取出的内容不是操作数,也不是下一条要执行的指令,而是操作数的地址或指令的地址。这种寻址方式称为间接寻址,简称间址。间接寻址有一次间址和多次间址两种情况,大多数计算机只允许一次间址,图 2(a) 与 (b) 分别表示寄存器一次间址和存储器一次间址,图 2(b) 取操作数需访问存储器两次。



(a) 通用寄存器间址



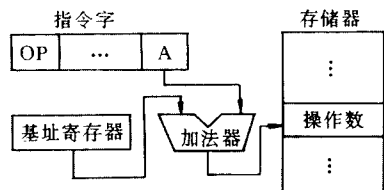
(b) 存储器间址

图 2 间接寻址

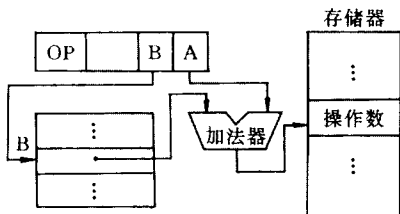
(4) 基址寻址

在中央处理器中设置一个专用的基址寄存器,或由指令指定一个通用寄存器作为基址寄存器,操作数的地址由基址寄存器的内容和指令的地址码 A 相加得到,如图 3 所示。在这种情况下,地址码 A 实际上是一个位移量。

基址寄存器主要用于为程序或数据分配存储区,或扩大地址码长度。通常基址寄存器中的内容只能由系统程序设定(通过特权指令实现),不能被用户程序所修改,因此确保了系统的安全。



(a) 专用寄存器



(b) 通用寄存器

图3 基址寻址

(5) 变址寻址

操作数地址由变址寄存器内容和指令的地址码A相加得到,如图4所示。图中 x 为变址寄存器在通用寄存器中的编号(即地址)。当计算机中还没有基址寄存器时,在计算操作数地址时还要加上基址寄存器内容。变址寄存器的内容是经常改变的。图中所示为处理数组的情况(x 由 $1 \rightarrow m$)。

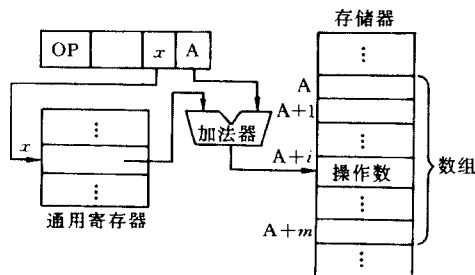


图4 变址寻址

(6) 立即数寻址

指令所需的一个操作数由指令的地址码部分直接给出,称为立即数(或直接数)寻址。这种寻址方式的特点是取指令时,同时取出一个操作数,提高了指令的执行速度,但只适用于操作数固定的情况,且字长较短,通常用于给某一寄存器或存储器单元赋初值或置常数。

(7) 相对寻址

把程序计数器PC的内容(即当前执行的指令的地址)和地址码部分给出的位移量相加作为操作数地址或转移指令的转移地址,称为相对寻址。转移类指令一般都用相对寻址方式,以适应浮动程序的需要。由于转移地址不是固定的,而是随着PC值的变化而变化,并且总是和当前指令地址(PC)相差一个固定的位移量,因此无论程序装入存储器的任何地方,均能正确运行。

有些计算机的程序和数据是分开存放的,在这种情况下,不能用相对寻址方式来确定操作数地址。

(8) 隐含寻址

指令的某个操作数或操作数地址隐含在某个寄存器或指定的存储器单元中,指令不必直接给出这一操作数或操作数地址,从而可以缩短指令的长度。这种方式在微处理器中普遍采用。

(9) 堆栈寻址

指令的一个操作数在栈顶,栈顶地址由一个专用的堆栈指针寄存器指出,这时指令就不必给出这一操作数或操作数地址。当取出操作数后,应修改堆栈指针,使栈顶的下一个单元成为当前栈顶。

以上这几种寻址方式可以适当组合起来使用。

对于二地址指令或三地址指令中的另一操作数和目的地址,可采用类似方法为每个地址码各自增设一个寻址方式字段。在一条指令中可同时存在多种寻址方式或采用同一种寻址方式;也可以作某些约定予以限制和简化。

假如用户用高级语言编程,则不必考虑寻址方式,因为在将高级语言程序转换成机器语言程序的编译程序中,已经考虑了寻址方式。假如用户用汇编语言编程,应对寻址方式有确切的了解,才能编出正确而又高效率的程序。

复杂指令集计算机采用了比较复杂和多样化的寻址方式,一般用高级语言编程,编译成机器语言程序后才能在计算机上运行,而**精简指令集**计算机用到的寻址方式种类则是不多的。(王爱英)

xunlian fangzhenqi

训练仿真器 (training simulator) 使用计算机仿真技术对人员进行训练的一种专用设备。它由3个部分组成:

(1) 计算机系统 它完成模型运算、数据转换、通信以及各种数值计算或符号处理的工作。

(2) 训练台 它为被训练者提供一个与真实系

统相类似的环境,如飞机座舱、电站控制室等。

(3) 教员台 它使教员能设置训练开始时的初始条件,给定训练科目,并记录训练结果,显示学员的训练过程,从而可对训练成绩进行评定。

凡是需要一个或一组熟练人员进行操纵、控制、管理和决策的真实系统,都需要对人员进行训练。对于复杂的系统,采用训练仿真器进行训练不仅能降低成本,提高安全性,而且能方便地安排训练计划,提高训练的效率和质量。有些训练(如危险事故处理)不可能在真实系统上进行却能在训练仿真器上多次重复地进行,因此训练仿真器具有极高的经济效益与社会效益。

20 世纪 40 年代,美国为训练空军飞行员首先研制成飞行训练器,当时的模型计算装置是机电的,训练的科目也极为有限。经过 50 多年的发展,现代训练仿真器不论是在技术水平上,还是在训练的内容上都发生了巨大变化。最新的飞机训练仿真器,采用并行处理计算机系统,视景系统则采用计算机图像技术,90%左右的飞行训练科目都可在训练仿真器上完成。

训练仿真器已应用到很多领域,一般分为以下 3 类:

(1) 载体操纵型 如飞机训练仿真器、舰船训练仿真器、火车汽车训练仿真器等。它以训练操纵

这些运载工具的人员为主要目的。

(2) 过程控制型 如电站训练仿真器、化工系统训练仿真器等。它以训练控制这类复杂系统的主控人员为主要目的。

(3) 博弈决策型 如飞机格斗训练仿真器、战术战役训练仿真器、调度管理训练仿真器等。它以训练军事指挥员、经理调度员以及其他各类决策人员的决策能力为主要目的。

训练仿真器的共同关键技术是:①研究与开发适合于训练要求的数学模型,并建立一套先进的建模环境,以便缩短训练仿真器的研制周期;②各种环境仿真的技术,如视景、音响仿真、具有加速度、不平衡运动、失重状态、运动感觉、力感觉的仿真等;③多功能、智能化教员台。

训练仿真器中广泛应用了计算机技术,特别是实时仿真技术,比如,利用计算机图形与图像技术(参见计算机图形学、数字图像处理)产生真实环境的景象;利用数据库管理各种模型并建立相应的模型库;利用人工智能技术对训练结果进行评价等。近年来,虚拟现实技术的发展更加提高了训练仿真器的逼真度,它使被训练者与由计算机产生的“虚拟环境”更加紧密地融合在一起,使被训练者有一种完全身临其境的真实感。

(熊光楞)

Y

yanyu(yuyin) hecheng fangfa

言语(语音)合成方法(method of speech synthesis) 用机械、电子、计算机等合成言语的方法。从采用的合成技术,可分为发音参数合成、声道模型参数合成和波形编辑合成;从合成策略上,可分为频谱逼近和波形逼近。

参数合成方法通常基于一定的语音产生模型,以共振峰合成为代表。它的优点是占用的存储空间小,与语音编码相结合时数码率较低,同时合成语音具有较高可懂度,并能够较灵活地控制合成语音的音色。参数合成的主要缺点是合成语音的自然度较低。参数合成方法有发音器官参数语音合成和声道模型参数语音合成两种。

发音器官参数语音合成 这种方法直接对人的发音过程进行模拟。它定义了唇、舌、声带的相关参数。如唇开口度、舌高度、舌位置、声带张力和肺气压等。由这些发音参数估计声道截面积函数,进而计算声波。这是对人发音过程的直接模拟,有可能产生逼真的语音。但由于人发音生理过程的复杂性,理论计算与物理模拟之间的差异,合成语音的质量暂时还不理想。

声道模型参数语音合成 这种方法基于声道截面积函数或声道谐振特性合成语音,如共振峰合成、线性预测系数(LPC)合成。较为著名的共振峰合成器是 MIT 教授 D. Klatt 设计的串并联混合型共振峰合成器。他用串联通道产生元音和浊辅音;并联通道产生轻辅音。还可以对声源做各种选择和调整,以模拟不同的噪音。在此基础上开发的 DEC Talk 英语文语转换已广泛应用。国内外已有不少基于参数合成技术的语音合成系统。这类系统需要的存储量低,音质适中,易于实现韵律修改。

波形编辑合成 通过对来自自然语音的语音基元进行拼接的方式产生合成语音,具有较高的自然度。它直接把语音波形数据库中的波形拼接在一起,输出连续语流。这种语音合成技术用原始语音波形替代参数,而且这些语音波形取自自然语音的词或句子,它隐含了声调、重音等细微特性,合成的语音清晰自然。其质量普遍高于参数合成。然而,

任何一个语音单元的声学特性,会随着语音环境的变化而改变。那么,如何收集和选择适当的语音单元呢?如果找不到合适的语音单元,怎么办呢?20 世纪 80 年代末 E. Moulines 和 F. Charpentier 提出基于时域波形修改的语音合成算法 PSOLA。PSOLA 就是基音同步叠加。它把基音周期的完整性作为保证波形及频谱平滑连续的基本前提。该算法按以下三步实施:它以基音周期的整数倍为窗长,对原始波形进行分析,产生中间表示;然后对中间表示进行修改;将修改过的中间表示重新合成为语音信号。由于修改的参数不同,又分为时域 TD-PSOLA、频域 FD-PSOLA 和线性预测 LP-PSOLA。该方法较好地解决了语音拼接中的问题,已经成功地应用于文语转换(TTS)系统中,现已有英、日、德、法、汉语等多种语言的系统问世。

参考文献

1. Eric Moulines and Francis Charpentier. Pitch-Synchronous Waveform Processing Techniques for Text-to-Speech Synthesis Using Diphones. *Speech Communication*, 1991: 453 ~ 467
2. Dennis H Klatt and Laura C Klatt. Analysis, Synthesis, and Perception of Voice Quality Variations among Female and Male Talkers. *J. Acoust. Soc. Am.*, 1990, 87(2): 820 ~ 857 (蔡蓬红)

yanyu(yuyin) hechengqi

言语(语音)合成器(speech synthesizer)

能将数字代码序列转换为言语信号,并能将言语信号转换为数字代码序列的大规模集成电路。又称**语音合成器**。在计算机应用中是指具有言语录制和回放功能以及言语合成功能的集成电路。

图 1 为 UM 5100 言语合成器内部结构及其应用框图。言语合成器可以通过外接静态随机存取存储器(SRAM)作为言语记录及回放,也可通过外接可擦编程只读存储器(EPROM)或只读存储器(ROM)用作言语回放。在言语记录方式下,话筒输入的言语信号经外接的低通滤波器和放大器滤波放大,再经合成器进行数据调制后产生串行脉冲信号。

串行脉冲信号通过串并变换成为 8 位并行信号,再送入三态数据总线缓冲器。由地址总线发生器向外接 SRAM 送地址信号($A_0 \sim A_{14}$),在写数信号控制下将三态数据总线缓冲器数据 $D_0 \sim D_7$ 写入 SRAM。当选择言语回放方式时,SRAM 输出数据 $D_0 \sim D_7$ 送入三态数据总线缓冲器,经串并变换器将 8 位并行信号变换为串行信号,并经数据调制器译码(解调)还原成言语信号。最后经过功率放大推动扬声器发声。

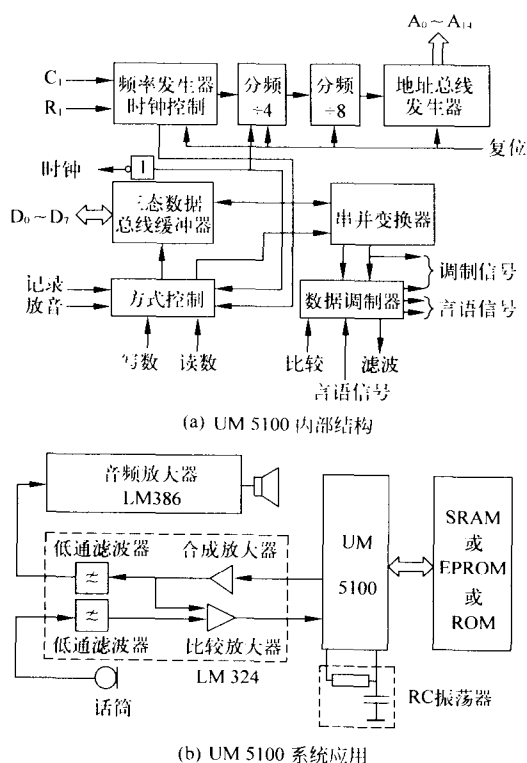


图 1 UM 5100 言语合成器结构及应用框图

言语合成器的具体结构因所用言语处理方式不同而异,因而其结构有多种,它们的合成原理、合成内容、合成时间、比特率及声音质量等也都各有特色。言语合成方法主要有 3 种:①波形编码方式,用固定采样周期把言语信号转换为数字代码序列放入存储器中,需要时读出存储内容,并转换为模拟信号;②分析合成方式,基于分析与模拟人的发音机理,从人的言语中提取与言语参数有关的特征参数进行言语合成;③规则合成方式,以音素或音节为单位进行言语合成。

近年来,除发展单独的言语合成器(芯片)外,

出现了中央处理器(CPU)与言语合成芯片相结合的结构。美国 TI 公司的单片言语合成器 TSP 50C1X 和 TSP 504X 系列即属此类产品。它采用线性预测编码(LPC)技术,以低数据率合成优质言语,因而可节省存储空间。单片言语合成器将 LPC 格形滤波器、8 位 CPU、程序 ROM 和言语数据 ROM、脉宽调制数模转换器集成在同一个芯片上,它有可编程数据处理的能力,因而具有高度灵活性。

参考文献

1. Italiano P, Ponte G, Sartori M. An Integrated High Quality Speech Synthesizer Based on LPC Techniques. IEEE Transactions on Consumer Electronics, 1985, CE-31:501~504

2. 方建淳编著. 语音合成技术与单片微机综合系统. 北京:北京航空航天大学出版社,1993

(时万春)

yanyu(yuyin) shibie de tezheng chouqu
言语(语音)识别的特征抽取(feature extraction of speech recognition) 用信号处理技术提取语音信号中可供相互区分的特征参数。特征参数大致分为两类,一类反映语音信号的频谱包络特性,即声道响应特征,如共振峰参数、线性预测系数、倒谱系数等;另一类反映语音的时域特性,如能量、音调等。特征参数选取对识别性能影响很大,目前用得最多而又有效的参数是倒谱系数。也有用快速傅里叶变换(FFT)系数和带通滤波器组的输出作为特征参数。能量是语音信号时域特征,将它用于识别系统,可使性能得到改善。倒谱系数和能量的变化,即它们的动态特性是语音信号的重要特征之一,在识别系统中起重要作用,它和静态特性在很大程度上是无关和互补的,有人建议使用倒谱的回归系数作为动态特性。选用什么特征参数要有利于识别,即在高维特征空间中要易于将不同的识别音区分开。许多研究表明,频域特征参数在频率维作非线性校正,使之符合人耳分析频率的特性,能给识别带来好处。

近年来的研究表明,采用听觉特性作频率分割 Mel 刻度倒谱(MFCC)是较好的识别特征,相比其他参数有较高的识别率。

人们发声器官的视觉特征,主要指口唇形状的特征,是语音识别的重要辅助特征,特别是在噪声环境中将语音的听觉和视觉特征结合起来,可以提高语音识别率。

参考文献

1. 陈永彬,王仁华. 语言信号处理. 合肥: 中国科学技术大学出版社, 1990
2. Stork D G. et al. Neural Network Lipreading System for Improved Speech Recognition. In: Proceeding of International Joint Conference on Neural Network. 1992, 2: 289 ~ 295 (莫福源 方棣棠)

yanyu(yuyin) shibie zhong de yuyan moxing
言语(语音)识别中的语言模型(language model of speech recognition) 言语(语音)识别中,语音单元被识别后产生合理语句的规则集合。在言语(语音)识别中按语言模型利用语言学和语法知识从识别的语音中(可能有几个候选音,它们有不同的概率)挑出正确的字或词来,使语音输入的句子(语音串)变成有正确意义的文字。用语言模型可大大减少识别搜索运算量 and 提高识别率。以 W 表示待识别的词或句子,以 A 表示待识别语音的声学信号。用 $p(A/W)$ 表示,说 W 的条件下产生 A 的概率,不同的 W 产生 A 的概率不同;用 $p(W)$ 表示话语中 W 出现的概率。在识别过程中,检测到一个确定的声学信号 A 之后,查 W (多个)出现 A 的概率 $p(A/W)$,再和 W 出现的概率 $p(W)$ 相乘。比较不同的 W 有不同的乘积,将乘积最大的那个 W 就作为挑出词,即识别结果。用公式表示为

$$\hat{W} = \arg \max_{\#} [p(W)p(A/W)] \quad (1)$$

公式中的 $p(A/W)$ 是由语言模型匹配得到的概率, $p(W)$ 是由语言模型决定的概率。建立语言模型有两种方法:一种是句法-语义分析方法。这是基于知识规则的方法,需要有庞大的专家知识库,规则十分繁复。一些特例不在规则之内,也许会引起错误。这种方法对一定应用范围的语音理解系统比较有效,但对无应用背景的普适性理解系统,则因知识和规则太多、太繁杂而使系统过于庞大。采用语言模型后,通过选择合法句子,还可将其中个别错误的词改正过来。在建立普适性的识别理解系统情况下,例如用汉语单音节或词汇呼入任意中文文本时,上面基于句法-语义的语言模型将因知识库和规则过于庞大而产生困难,可代之以另一种用语言统计模型的办法。这是用大量语料,包括政治、经济、文艺、体育、科技等各个领域的语料来统计词的前后搭配概率,利用统计所得概率去引导或控制语音识别匹配范围和判决。例如,在已有前 N 个词 W_1, W_2, \dots ,

W_n 情况下出现词 W 的概率,称为 N 词语言模型。这样的模型因参数太多而使模型极为复杂,常用前一个词或前两个词的简化模型。用前一个词的统计语言模型称双词语言模型,用前两个词的模型称为三词语言模型。语言统计模型不但将建立模型的大量工作交由计算机去承担,而且和语音识别匹配模型连接更为直接和方便,为许多语音识别理解系统所采用。显然,利用知识和统计这两种方法建立的语言模型是相互重叠、相互补充的。

语言统计模型基本上是从历年报纸上大量新闻中统计得到的。对某一特定主题(例如医药、农业、金融等)并不完全适用。针对某一特定主题,计算机自动进行在线语言模型自适应是改善语言模型,提高音字转换正确率的重要研究方向。

参考文献

1. Waible A, Lee K F. Readings in Speech Recognition. San Mateo: Morgan Kaufmann Publishers Inc., 1990
2. 吴振清,郑方等. 一种在线递增式语言模型自适应方法. 见:第6届全国人机语音通信学术会议, 2001 (莫福源 方棣棠)

yanse fuzhi

颜色复制(color reproduction) 将扫描仪、数码相机等设备获取的和(或)计算机处理得到的数字彩色图像在打印机、彩色印刷机等硬复制设备上真实地再现的过程、方法和技术。又称颜色重现。

扫描仪、数码相机获取的或者计算机生成的彩色图像,经过计算机编辑、排版,并通过彩色显示器获得了满意的彩色效果之后,为了在彩色硬复制设备上获得颜色效果完全相同的输出,还需要进行如下处理:

(1) 色域调整与色彩匹配 可见光中包括几百万种不同的颜色。而显示器、扫描仪、彩色打印机等各有自己的颜色范围(称为色域),其中使用 CMY 三原色的彩色打印机是色域最窄的一种(图1)。在计算机屏幕上看起来极佳的 RGB 颜色,有时会因为转换到 CMYK 模式进行彩色打印而变得模糊不清。这是因为显示器上的图像所包含的颜色无法全部在 CMYK 模式下复制。为此,需要进行色域调整与色彩匹配,让人的眼睛在具有不同色域的不同设备上,所看到重新产生的颜色尽可能一致。

(2) 分色 计算机进行排版和图像处理时,彩

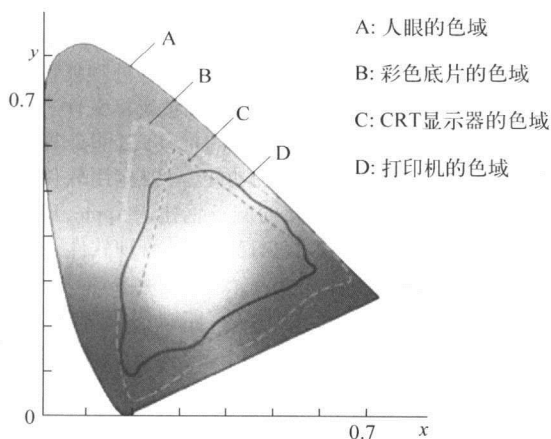


图1 不同设备的色域

色图像采用 RGB 颜色模型表示。而硬复制输出设备一般都使用 CMYK 模型。分色就是把彩色图像分解成青、洋红、黄、黑 4 种颜色的灰度图。它首先把彩色图像中的各种颜色由 RGB 颜色模型转换成 CMYK 颜色模型,然后再将彩色图像储存为青、洋红、黄、黑 4 种颜色的灰度图。

在实际打印或印刷过程中,加入黑色颜料的目的是为了弥补 CMY 三原色形成的黑色不够黑这一缺陷。此外,直接用黑色颜料形成不同的灰度级而不用 CMY 三原色组合产生,也可节约大量的彩色颜料。将 RGB 颜色转换为 CMYK 并不是惟一的,通常的做法是先决定黑色成分,再计算 CMY 三原色的数值。

如果在原有 CMYK 颜色的基础上再加入淡蓝绿色、淡红紫色和其他颜色,即使用六色或七色的彩色喷墨打印机输出彩色图像,其分色处理的过程也是非常复杂的。

(3) 伽玛校正 因为人眼对灰度等级的灵敏度与实际设备输出的灰度级并不成线性关系,而是指数关系,为了得到较好的视觉效果,在实际输出时,通过伽玛校正对每一个颜色分量的灰度值调用传递函数进行修正。

(4) 挂网 分色后的图像是若干个具有连续色调的灰度图。在彩色印刷中,每次印刷时只能使用一种油墨,且油墨的浓度保持不变。为了在印刷时获得连续色调,需要对灰度图进行挂网处理。挂网也叫加网,它把连续色调的图像分解成网点来表示,用网点的大小和疏密反映图像颜色的深浅层次。其原理是基于人的如下视觉特性:从一定距离观察一

块小面积区域(如 $0.02\text{mm} \times 0.02\text{mm}$)中的图像时,区域中网点较大看起来就暗,网点较小看起来就亮;网点稠密看起来暗,网点稀疏看起来亮。

按照网点形成的方法,挂网技术主要分为调幅挂网和调频(随机)挂网两种。调幅挂网是通过改变网点大小来实现不同灰度的方法,网点大的地方颜色暗,网点小的地方颜色亮。调频挂网是在随机图案中印刷相同大小的网点,通过改变网点的稀疏(密度)来实现不同灰度的方法。这种方法网点的放置无规则,印刷品不会产生纹路。按照由软件还是硬件完成挂网处理,分为前端挂网和后端挂网:前端挂网指软件挂网,它是在图像输出之前先做挂网处理,然后将挂网处理后的图像数据存储在磁盘或光盘上,供输出胶片时调用;后端挂网也叫硬件挂网,它在输出胶片的同时,由专用硬件栅格图像处理器(RIP)对图像实现高速挂网。

颜色复制是电子印前处理系统的主要功能之一。通用的印前处理系统除了计算机设备与相关软件之外,还需要配置高精度激光照排机(输出制版用的胶片)和栅格图像处理器(RIP)。专用的系统称为电子分色机(简称电分机),它是一种彩色制版设备,采用光电扫描方法获取原稿的图像信息,借助于电子技术和激光技术,能够从彩色图像直接制成经过彩色校正、层次校正、黑版计算、底色去除、细微层强调、挂网处理等可满足各种制版条件和印刷条件的分色胶片,在彩色印刷行业得到了广泛的应用。

完成颜色复制任务的软件是颜色管理软件,它通过使用类似于 CIE $L^*a^*b^*$ 的与设备无关的颜色模型来调校不同设备之间的颜色。

参考文献

多妮,奥奎恩等. 数字印前大全. 北京:机械工业出版社,1997 (张福炎)

yanse guanli

颜色管理(color management) 使计算机中相同的颜色数据,不管用什么系统和(或)设备输出,都能获得相同彩色效果的方法和技术。

随着计算机及外部设备的发展,出现了彩色桌面出版系统,并得到了迅速的普及和发展。彩色出版系统具有开放性,系统组成灵活,不同厂商生产的各种设备可以集成在一个系统中。然而经常有这样的情况:在显示器上看到的颜色与打印出来的颜色差别很大;不同的扫描仪扫描同一幅图像,会得到不

同颜色的图像数据;不同型号的显示器显示同一幅图像,也会有不同的显示效果;印刷中使用不同的油墨,得到的颜色效果也不一样;甚至用同样的油墨而使用不同的纸张,也会导致不同的结果等。产生这些现象的因素很多,其中还包括人的因素,因为每个人对颜色的感觉是不同的。因此,彩色出版系统必须进行颜色管理,才能有效地解决这些问题。

为了使相同的颜色数据,在所有设备上产生相同的彩色效果,颜色管理需要完成的任务包括颜色校正、颜色定标和颜色转换。颜色校正是使各有关设备达到规定的标准参数;颜色定标就是确定输入及输出设备的颜色范围或可再现的颜色是哪些;颜色转换是将图像从一个设备的颜色空间转换到另一个设备的颜色空间。不同的设备(输入、显示、打印)表示颜色的方法不同,各自所能生成颜色的色域也不相同,为此必须建立描述各个设备颜色范围的数据文件,这种文件称为设备的颜色特性文件。颜色管理软件的目的,就是通过对所有设备的管理,补偿和控制这些设备间的差别,以得到精确的可预测的颜色。

颜色管理软件中需要使用一个与设备无关的标准颜色空间,一般使用 CIE $L^*a^*b^*$,它也叫做基准颜色空间,在转换过程中起着连接的作用。当扫描一幅图像时,颜色管理软件使用储存在扫描仪颜色特性文件中有关扫描仪的信息,将扫描仪的 RGB 图像转换到 CIE 空间中。然后,再使用存储在显示器颜色特性文件中有关显示器的信息,将 CIE 空间的图像转换到显示器颜色空间,然后在显示器屏幕上显示。同样,当打印输出图像时,颜色管理软件再次使用两个设备的颜色特性文件中的信息,通过 CIE 作为基准空间,将显示器 RGB 颜色空间的图像数据转换成打印机的 CMYK 颜色空间,如图 1 所示。

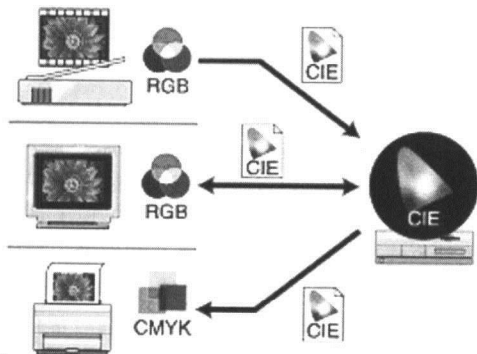


图1 颜色管理

颜色管理中使用的设备颜色特性文件,是由颜色专家使用专门的仪器通过测量若干典型应用中每个设备的颜色特性而得到的。一些著名的颜色管理软件开发商可以提供许多设备的颜色定标文件。

由于国际颜色联盟(ICC)的推动,目前计算机系统均已内置颜色管理功能,只要应用软件及硬件支持,便可获得准确的颜色输出。Apple, EFI, Microsoft 等不少厂商都有自己的颜色管理软件。

参考文献

张桂兰. 浅谈色彩管理系统. 印刷技术, 2000, 1: 18 ~ 22 (张福炎)

yanse moxing

颜色模型(color model) 颜色信息的描述与定量表示方法。又称颜色空间或颜色系统。

人的视觉器官(眼睛)有感觉外界物体的形象、光和颜色的功能。人的色觉可以分辨数千种颜色。

有了光的照射,才有被人眼看到的颜色。黑暗中是无颜色可言的。光是一种能对人眼视觉神经起刺激作用的电磁辐射。电磁辐射的波长范围很广,可见光的波长范围一般指 380 ~ 780nm,称为可见光谱。在可见光谱范围内,不同波长的辐射使人们感觉到不同颜色,一般来说,700nm 为红色,580nm 为黄色,510nm 为绿色,470nm 为蓝色,400nm 为紫色。

颜色从产生的原理讲,可分为光源色和物体色两大类。太阳光、星光、各类灯光等,它们自身会发出不同颜色的光。太阳光习惯上被称为白光,它是由多光谱混合组成的复色光。除了光源色之外,其他的都属于物体色。物体色的特点是:①没有光照就没有颜色,在不同的光照条件下呈现不同的颜色。②当光照到一个物体上后,一部分光会被这个物体反射(或透射),反射(或透射)光的特性,决定了看到的物体颜色,所以物体表面的特性是决定它的颜色的重要因素。③物体表面对光的反射与照射光的特点或角度有关。光束从什么角度照射,眼睛从什么角度观看,都会影响到所看到的颜色。

除了光源和物体之外,人对颜色感觉的形成还有两个要素:眼睛和大脑。由于人眼的透镜作用,使外来物体的光线呈现在视网膜上。视网膜在眼球壁的内层,它是眼球的感光部分,是一个透明薄膜,其中有视觉感光细胞——锥体细胞和杆体细胞。锥体细胞能够分辨颜色和物体的细节,杆体细胞适合于弱光视觉,但不能分辨颜色和细节,它们在接受光色刺激后,将这个刺激转化为神经冲动,经过传递最后

传到大脑枕叶皮层的高级视觉中枢,人就产生了物体的大小、形状和颜色的感觉。值得指出的是,不同的人,由于民族、性别、年龄、文化程度、疲劳状况及职业等因素的不同,对颜色感觉会略有差别。

总之,光的存在,物体的表面特性和人眼的视觉功能,是形成颜色的必要因素。

颜色是人们的主观感觉,日常生活中往往用粉红、大红、浅蓝、深蓝等定性地描述各种颜色。随着技术的发展,服装、印染、美工、彩印、电视等行业要求采用客观的定量的方法来精确地描述颜色,为此必须建立颜色的表示方法,颜色的表示方法就是颜色模型。

颜色模型有多种,不同的颜色模型有不同的应用,它们可以相互转换。下面是几种常用的颜色模型。

(1) RGB 模型 即红、绿、蓝三原色模型,它是一种加色模型,即通过不同亮度的红、绿、蓝三原色的叠加可以生成各种不同的颜色。RGB 模型中 3 种原色分别有 256 个亮度,它们的所有不同亮度的组合便形成了色彩丰富的颜色空间(图 1)。RGB 模型适用于彩色电视机、计算机显示器等主动发光的设备。

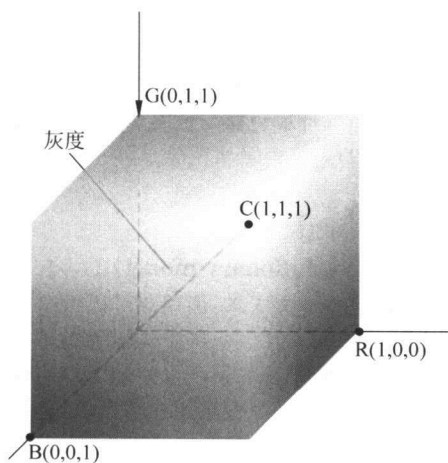


图 1 RGB 模型

(2) CMY(CMYK)模型 该模型使用的 3 种原色分别是青(C)、品红(M)和黄(Y),其视觉效果等于从白色光中分别减去红(R)、绿(G)、蓝(B)3 种颜色,因此称为减色模型。CMY 模型适用于彩色印刷、印染和彩色打印机,因为布料上或纸上的颜料是通过光线反射的原理再现颜色的。

减色模型理论上可以用青、品红、黄三原色合成所有的颜色,但实际利用它们来合成黑色时,往往由于颜料不纯或油墨不易完全混合而不能生成真正的黑色。因此,通常用真正的黑色油墨来取代颜色中的黑色成分,以获得更真实的效果。这时,CMY 模型就演变为 CMYK 模型。

(3) HSV 模型 HSV 模型比较符合人们对颜色的直观感觉,它使用色调(H)、纯度(S)和明度(V)3 个参数来描述颜色。其中 H 表示色彩信息,即颜色光谱的位置,用角度来表示,红、绿、蓝分别相隔 120° ,互补色分别相差 180° ;纯度 S 也称为色彩的饱和度,它是一个比例值,范围从 0 到 1,表示所选色彩的纯度和该色彩最大纯度之间的比率;V 表示色彩的明亮程度,范围从 0 到 1。

HSV 模型用六面锥体(或圆锥体)来表示,如图 2 所示。V 轴是明度值,顶面是 $V=1$ 时(最大亮度值)的各种颜色;H 是绕 V 轴旋转的角度。 0° , 120° , 240° 分别表示红、绿、蓝;S 表示离开轴心的距离,距离越远表明饱和度越高。

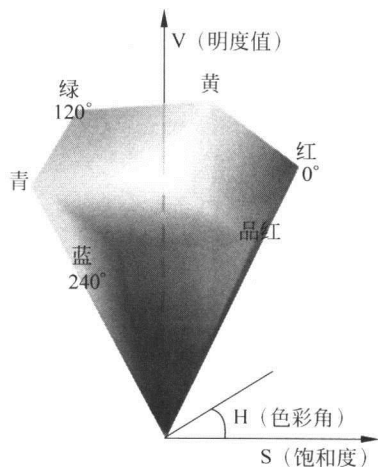


图 2 HSV 模型

(4) YUV(YIQ)模型 在广播电视中,通常总是把 RGB 模型表示的彩色图像信号变换成为用 YUV(或 YIQ)模型表示后才进行远距离传输,被电视机接收后再恢复为 RGB 信号在显像管上进行图像显示。YUV(或 YIQ)模型中的 Y 表示亮度信号,另外两个分量分别表示两种色度信号。由于人们对色度变化的敏感程度远低于亮度的变化,因此两个色度信号的传输可以比亮度信号少占用一些频带,以节省频率资源,同时也保持了与黑白电视的兼容。

其中 YUV 模型适用于 PAL 和 SECAM 彩色电视制式, YIQ 模型则适用于 NTSC 电视制式。

(5) Munsell 颜色系统 这是美国画家 A. H. Munsell 所创立的一种颜色模型。该模型按颜色的 3 种基本属性描述颜色, 这 3 种属性是: 色调、色度和亮度。色调表示红、黄、绿等颜色特征, Munsell 模型中有 5 个主色调(红、黄、绿、蓝、紫)和 5 个中间色调, 每一种色调又细分为 10 级; 色度指色调的纯洁程度(饱和度), 色越纯, 在人的感觉上就感到越浓艳, 每一种色调的色度分级不完全相同; 亮度指相对明暗的特性, 分成 11 级(图 3)。Munsell 颜色系统是目前国际上广泛采用的颜色分类标准, 在标定物体表面颜色(包括颜料、油墨、印刷品等)时使用。

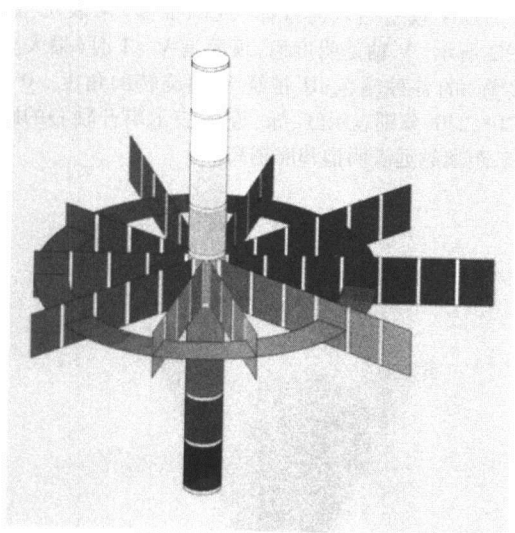


图 3 Munsell 颜色系统

(6) CIE-XYZ 模型 上述颜色模型的缺点是它们不能表示出所有可见的颜色。为了用三原色定义出所有的颜色, 国际照明委员会(CIE)定义了 3 种假想的颜色 X, Y, Z 作为标准原色, 其他所有的颜色 Cr 都可以用它们的组合来表示, 即 $Cr = aX + bY + cZ$, 这就是 CIE-XYZ 颜色模型。

令 $x = X/(X + Y + Z)$, $y = Y/(X + Y + Z)$, 这样 x, y 的值就只依赖于颜色的色彩和纯度, 而与颜色的亮度无关, 参数 x, y 称为色度。图 4 所示的 CIE 色度图中, 马蹄形曲线上的点是光谱颜色, 即纯彩色, 它们的纯度最高。曲线包围的区域表示自然界中所有的颜色, C 点为白色, 越靠近 C 点颜色的纯度就越低。

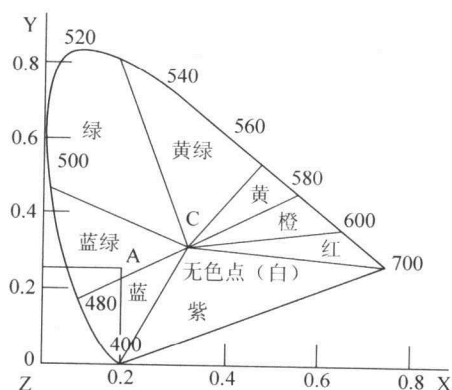


图 4 CIE 色度图

为了使颜色的设计和复制更精确, 人们希望颜色空间中不同位置、不同方向上相同的距离在视觉上最好能对应相同的色差, 把易测的空间距离作为颜色感觉差异的度量, 减少由于空间的不均匀而带来颜色复制的误差。为此, CIE 对 CIE-XYZ 颜色模型进行了数学变换, 1976 年推荐了新的颜色模型, 即 CIE1976LAB(或 $L^*a^*b^*$) 模型, 它适用于一切光源色或物体色的表示与计算, 已为世界各国正式采纳, 成为国际通用的测色标准。

参考文献

- 王大珩, 王继宪, 滕秀金, 胡维生, 王恒. 颜色的产生. 光明日报·科技周刊, 2002-02-22
- 胡成发. 印刷色彩与色度学. 北京: 印刷工业出版社, 1993

(张福炎)

yanhua moxing

演化模型 (evolutionary model) 一种主要针对事先不能完全定义需求, 但用户可以给出待开发系统的核心需求, 并且当看到其实现后, 能够有效地提出反馈, 以支持系统的最终设计和实现的软件开发模型。软件开发人员根据用户的需求, 首先开发核心系统。当该核心系统投入运行后, 用户试用之, 完成他们的工作, 并提出精化系统、增强系统能力的需求。软件开发人员根据用户的反馈, 实施开发的迭代过程。每一迭代过程均由需求、设计、编码、测试、集成等阶段组成, 为整个系统增加一个可定义的、可管理的子集。如图 1 所示。

如果在一次迭代中, 有的需求不能满足用户的要求, 可在下一次迭代中予以修正。

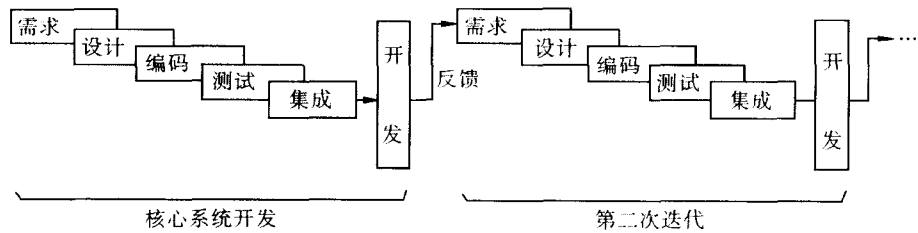


图1 演化模型

演化模型在一定程度上减少了软件开发活动的盲目性。
(王立福)

yanyi shujuku

演绎数据库 (deductive database) 具有演绎推理能力的数据库。

在传统数据库（主要是关系数据库）上增加一组规则以及建立一个推理机制，使之能在现有数据库的数据上演绎出新数据的功能。

演绎数据库的推理机制是演绎数据库的核心，它的主要内容是一些演绎推理算法，其中主要是递归查询算法。这些算法大体分为自顶向下和由底向上两大类。主要有自顶向下的 QSQI, QSQR, 及 QRGT 等算法以及由底向上的 Naive 算法与 Semi-Naive 算法。20 世纪 80 年代后期出现的 Magic Set 算法保持了自顶向下与由底向上的优点，并且避免了两者的缺点。此外，一些特殊的递归查询优化算法较大地提高了查询处理的效率，也受到了特别的重视，如线性递归查询优化算法、左线性递归查询优化算法、右线性递归查询优化算法及左-右线性递归查询优化算法等。

演绎数据库一般由外延数据库 EDB 与内涵数据库 IDB 两部分组成。其中 EDB 存储实际存在的数据称实数据，而 IDB 则是存储那些由规则经推理机制演绎而得的数据，它们并不实际存在于数据库中，仅仅是在需要时才通过演绎而得到，故称为虚数据。

由于 IDB 的存在使得演绎数据库能提供给用户的数据远远多于实际存在的数据，而所占用的空间则几乎与实数据所占用的空间一样，这就显示出演绎数据库的特有优势。

演绎数据库采用一阶谓词逻辑为表示工具，为适应数据库应用需要，对其作适当改造而形成一个以 Horn 逻辑为基础的 DATALOG 语言。DATALOG

语言由事实与规则两部分组成，分别用以表示 EDB 中结构与数据以及 IDB 中的规则。演绎数据库有两种理论基础，一种是基于证明论的演绎数据库，另一种是基于模型论的演绎数据库，两种不同基础的演绎数据库实现时采用不同的算法。

演绎数据库具有巨大的应用潜力，但是在实现中由于递归规则查询算法效率偏低而影响了它的使用与推广。目前较著名的演绎数据库系统有美国斯坦福大学研制的 NAIL! 系统，美国 MCC 的 LDI 系统，墨尔本大学研制的 Aditi 系统以及美国加州大学的基于 INGRES 的演绎数据库系统 POSTGRES。

20 世纪 70 年代后期，关系数据库技术与人工智能技术这两者的结合产生了演绎数据库，目前，演绎数据库的主要发展内容是算法研究产品研制及应用推广。

参考文献

Ullman J D. Principle of Database and Knowledge Base Systems. Vol. II, New York: Computer Science Press, 1989

(徐洁磐)

yanyi zonghe fangfa

演绎综合方法 (deductive synthesis method)

从给定的软件规约借助演绎推理综合出程序的方法。简称演绎方法。它将数学中的构造性证明与软件开发相联系，把开发步骤解释为证明步骤，从证明中抽取相应的程序。

主要工作 Z. Manna 和 R. Waldinger 的 Tableau 方法是一种较典型的基于演绎推理的方法。在 Tableau 方法中，把给定的逻辑公式分解成子公式后，对子公式加以证明。证明过程伴随程序的构造，证明的结束表示程序构造完毕。PRL 系统和 Nuprl 系统是 R. Constable 等人研制开发的基于 P. Martin-Löf 的类型理论的演绎综合方法，系统交互地进行定理证明，并从证明中抽取程序。

基本方法 从程序规约导出程序的过程可叙述为:假定欲构造的程序 F 的输入为 x , 满足给定输入条件 P , 程序的输出为 y , 并满足输出条件 Q , 则待综合的程序的形式规约:

$$F(x) \leq = \text{find } y \text{ such that } Q(x, y) \\ \text{where } P(x)$$

可转换为下述命题:

$$\forall x \exists y (P(x) \rightarrow Q(x, y))$$

如果能构造性地对该命题给出证明, 则可抽取出待综合的程序。程序既可在证明命题过程中逐步构造, 也可在证明之后再行抽取。

演绎过程 定理证明过程是一演绎过程, 它可以通过一生成结构来记录证明过程的每一步。定理证明本身并不能生成程序, 因此, 必须在证明过程中记录可能的输出项, 即为要构造程序的片段。当证明结束时, 可根据这些片断最终得到所期望的程序。

综合技术 基于定理证明技术的演绎综合最终将不可执行的规约转换为可执行的目标语言程序, 这表明, 演绎综合涉及到程序设计语言、程序设计技术等知识。这些知识通常用规则形式表述。例如, 循环、条件和递归是程序设计语言的基本构造, 演绎综合必须能通过施用相应规则对待构造的程序引入这些构造。主要规则有: ①转换规则: 逐次将转换规则应用到给定的规约, 在保持规约意义的前提下, 用目标语言的基本构造代替规约中的非基本构造。②条件形成: 某些转换规则要求某些条件在施用规则之前为真。当施用该规则而又不能证明或推翻前提条件时, 引进一个基于该条件的情况分析, 在最终的程序中形成一个条件表达式构造。③递归引入: 当发现一个新子目标恰好是顶层目标的实例时, 就可以直接引进递归调用。但是, 当子目标不是顶层目标的实例时, 通常必须构造一个新的过程(或子程序)来计算比此更一般目标的表达式, 而用这个新过程的调用来达到原来的目标。④推广: 在定理证明中, 推广技术已广为应用。用数学归纳法证明定理时, 往往有必要去证明一个更一般的定理。在程序综合中, 递归与归纳类似, 可试图构造计算一个更一般的目标的程序。因此, 要求递归调用足以达到所希望的子目标。

方法改进 为减小实现证明的复杂性, 综合较大规模和结构良好的程序, D. R. Smith 以分治法为基础设计了一个半自动化系统 CYPRESS, 它使用了比定理证明更一般的前件推导技术, 并用条件作为分解软件规约的手段, 而非直接将算法设计归结为

定理证明。为了进一步克服纯定理证明技术所带来的局限性, Z. Manna 等人放弃了纯定理证明方法, 而采用结构定理证明技术、数学归纳法和程序转换技术的演绎式程序综合方法。

参考文献

1. 徐家福, 陈道蓄, 吕建, 王志坚. 软件自动化. 北京: 清华大学出版社, 1994
2. Manna Z et al. A Deductive Approach to Program Synthesis. ACM Transaction on Programming Languages and Systems. 1980, 2(1): 90 ~ 121

(王志坚 张家重)

yangtiao hanshu

样条函数 (spline function) 一类分段(片)光滑, 各段(片)交接处具有一定光滑性的函数。样条函数的名称来源于船体放样时用于画光滑曲线的机械样条——弹性的细长条, 样条函数产生的背景是离散数据的拟合。高次多项式插值过程有数值不稳定的缺点, 而利用分段低次多项式插值, 则在分段处有一定光滑性、较好的稳定性和收敛性, 这种插值过程产生的函数就是(多项式)样条函数。样条函数的概念是美国数学家 I. J. Schoenberg 于 1946 年提出的, 但当时没有引起人们的重视, 随着科学技术和电子计算机的迅速发展及生产部门的需要, 20 世纪 60 年代中期, 它与计算机辅助几何设计相结合, 在外形设计方面得到了成功的应用。同时样条函数理论也逐步完善, 其应用亦逐渐扩充到各类数据的插值、拟合与平滑、数值微分与积分、微分方程的数值解法等方面。

n 次样条函数 给定区间 $[a, b]$ 上的一个分划 $\pi: a = x_0 < x_1 < \cdots < x_k < x_{k+1} = b$, 若函数 $S(x)$ 在区间 $[a, b]$ 上有 $n-1$ 阶连续导数, 且在每一个子区间 $[x_i, x_{i+1}]$ 上是 n 次多项式, 则称 $S(x)$ 是在 $[a, b]$ 上关于分划 π 的 n 次样条函数。 x_1, x_2, \cdots, x_k 称为样条结点。最常用的是 $n=3$ 的三次样条函数, 它的几何图形称为三次样条曲线。若在区间 $[a, b]$ 上给定函数 $f(x)$, n 次样条函数 $S(x)$ 满足插值条件 $S(x_i) = f(x_i) \quad i=0, 1, \cdots, k+1$, 则称 $S(x)$ 为 n 次插值样条函数。 $n=1$ 为分段线性插值; $n=3$ 为插值三次样条函数, 它有许多最佳性质。例如, 满足 $S'(a) = f'(a), S'(b) = f'(b)$ 的插值三次样条唯一存在, 且在一切满足上述插值条件的三次连续可微函数类中使 $\int_a^b [f''(x)]^2 dx$ 取极小值。关于插值三

次样条函数可参见插值。

插值三次样条的余项估计: 当 $f(x) \in C^4[a, b]$ 时,

$$\begin{aligned} \max_{a \leq x \leq b} |f^{(4)}(x) - S^{(4)}(x)| \\ \leq C_\alpha \max_{a \leq x \leq b} |f^{(4)}(x)| h^{4-\alpha} \end{aligned} \quad (\alpha = 0, 1, 2, 3)$$

式中 C_α 为与 $f(x)$ 无关的常数, $h = \max_{0 \leq i \leq k} (x_{i+1} - x_i)$ 。

n 次样条还可表示为

$$S(x) = \alpha_0 + \alpha_1 x + \cdots + \alpha_n x^n + \sum_{i=1}^k \beta_i (x - x_i)_+^n / n!$$

式中记号 $u_+^p = (\max(u, 0))^p$ 。这种表达式称为多项式样条。

n 为奇数的样条, 具有与三次样条类似的许多最佳性质。当 $n = 2$ 得到二次单结点样条, 即抛物线样条, 它是偶次样条的代表。二次样条有许多不同于三次样条的性质, 为了克服误差的传播放大, 用二次样条作插值时, 宜用样条结点的中点 $x_{i+\frac{1}{2}} = (x_i + x_{i+1}) / 2$ 作插值结点。

多项式样条即分段多项式, 可通过增加结点数或提高结点重数来增加样条空间的维数。它既有低次多项式的简单性, 又有相当的光滑性和灵活适应性, 还避免了高次多项式插值的不稳定性, 是函数逼近的重要工具。

多项式 B 样条 设实轴上的结点序列为

$$\begin{aligned} x_{-n+1} \leq \cdots \leq a = x_0 < x_1 < \cdots < x_{k+1} \\ = b \leq x_{k+2} \leq \cdots \leq x_{k+n} \end{aligned}$$

称 $B_{i,n}(x) = (x_{i+n} - x_i) [x_i, x_{i+1}, \cdots, x_{i+n}](t - x)_+^{n-1}$ 为 $[a, b]$ 上第 i 个 (以 $x_i, x_{i+1}, \cdots, x_{i+n}$ 为结点) $n-1$ 次 (n 阶) B 样条函数。各阶 B 样条函数由递推公式

$$B_{i,1}(x) = \begin{cases} 1, & x \in [x_i, x_{i+1}] \\ 0, & \text{其他} \end{cases}$$

$$B_{i,L}(x) = \frac{x - x_i}{x_{i+L-1} - x_i} B_{i,L-1}(x) + \frac{x_{i+L} - x}{x_{i+L} - x_{i+1}} B_{i+1,L-1}(x)$$

$$(i = -n+1, -n+2, \cdots, k; L = 2, 3, \cdots, n)$$

计算。 $n-1$ 次 B 样条函数有如下性质:

(1) 局部正支撑性 即当 $x \in (x_i, x_{i+n})$ 时, $B_{i,n}(x) > 0$; 否则, $B_{i,n}(x) = 0$ 。

(2) 叠加性 即对于任何两个结点 x_r 及 x_s , 当 $x_r < x < x_s$ 时, 有

$$\sum_{i=r-n+1}^{s-1} B_{i,n}(x) = 1$$

(3) 最小正支撑性 即

$$B_{i,n}(x) > 0 \quad x \in (x_i, x_{i+n})$$

$$(i = -n+1, -n+2, \cdots, k)$$

(4) 线性无关性 即 $[a, b]$ 上的 $n-1$ 次 B 样条函数 $B_{-n+1,n}(x), B_{-n+2,n}(x), \cdots, B_{k,n}(x)$ 线性无关。

区间 $[a, b]$ 上关于分划

$$\Delta: a = x_0 < x_1 < \cdots < x_{k+1} = b$$

的 $n-1$ 次多项式样条函数空间的维数是 $n+k$, 由 B 样条函数的线性无关性可知: 区间 $[a, b]$ 上的 $n-1$ 次 B 样条函数 $B_{-n+1,n}(x), B_{-n+2,n}(x), \cdots, B_{k,n}(x)$ 正好是该空间的一组基底。因此区间 $[a, b]$ 上关于分划 Δ 的任一 $n-1$ 次多项式样条函数 $S_{n-1}(x)$ 可以由 $\{B_{i,n}(x)\}_{i=-n+1}^k$ 线性表示, 即

$$S_{n-1}(x) = \sum_{i=-n+1}^k \beta_i B_{i,n}(x)$$

在 B 样条的基础上还发展了诸如切比雪夫样条、 L 样条或微分算子样条、指数样条等样条函数。三次样条是在小挠度的假设下得到的, 为了处理大挠度问题, 还建立了种种非线性数学模型, 发展了诸如圆弧样条和有理样条等, 它们在实际应用中也很

参考文献

Schmaker L. L. Spline function: Basic theory. New York: Wiley, 1981 (方遼)

yaogan xinxi chuli

遥感信息处理 (remote sensing information processing)

从接收和记录遥感器探测到的反映地物波谱特性的原始数据开始, 至回放出能客观反映地面或对象状况的图像, 所涉及的图像复原、几何校正、图像变换、图像增强、图像分类、统计分析、信息提取等技术和方法的统称。其目的是使图像变成便于理解和使用的形式, 或提取某些特征信息供进一步分析使用。处理手段包括解析和数字的计算机及光学处理方法等。处理所涉及的主要理论与技术包括空间定位、目标重建、影像匹配、模式识别、图像解译、数据库管理、空间信息理论、语义和非语义信息提取、知识工程、人工智能与专家系统及计算机视觉等。

遥感信息是由飞机、卫星等飞行器上的探测装置, 从空间不同高度探测和收集地表物体反射或辐射的电磁波信息。不同的卫星和遥感平台, 载有不同的传感器, 探测不同的目标物, 获取不同的信息, 分别有陆地、海洋、气象和地球资源卫星遥感信息以

及红外、微波、多光谱和高光谱遥感信息等。遥感信息是遥感的精髓,遥感的整个过程均围绕遥感信息而展开。

遥感的目的是为了得到研究对象的特征信息。这种特征信息通常是图像形式的遥感数据,具有有时相多、内涵丰富和信息量大等特点,例如,一颗卫星每天可取几千兆字节的遥感信息。因此,遥感信息处理主要是针对图像形式遥感数据的处理。它包括图像复原、几何校正、图像变换、图像增强、图像分类、信息提取等内容。

图像复原 在遥感信息处理中的图像复原是使卫星遥感数据能较真实地反映地球表面特征。一般复原过程包括退化效应模型化、复原处理器设计、空域或频域上执行运算等步骤。

几何校正 是对遥感图像的几何畸变进行校正处理。由于遥感器姿态角变化、扫描速度不稳定和地形起伏等因素,使图像产生几何形式或位置的失真。几何校正包括粗校正和精校正两种。前者是对图像的变形规律或遥感器在飞行时的位置和姿态进行模拟和解算,求出变换参数,再用这些参数改算所有点;而后者则通过采集地面控制点,建立校正多项式,求取各内插点的改正数,以达到校正目的。

图像变换 按一定规则将一幅遥感图像加工成另一幅图像的处理过程。变换方法有主成分分析、色度变换以及傅里叶变换等,还包括一些针对遥感图像的特定变换,如缨帽变换等。通过变换,增强所需的目标信息,使之便于识别、分析或作进一步处理。

图像增强 指应用计算机或光学设备改善遥感图像视觉效果,但没有增加信息量的处理。增强处理的内容包括反差增强和滤波。前者常用的方法有对比度扩展、彩色增强、多波段图像组合和变换,其目的在于改善图像上类别的判读效果;后者分为空间域滤波和频率域滤波,其目的是为了提取或抑制图像的边缘、细节特征或消除噪声等。

图像分类 指将图像中所含的多个目标物利用计算机进行识别和区分开的遥感信息处理方法。计算机分类的基本原理是计算图像上每个像元的灰度特征,然后根据不同的准则进行分类。遥感图像分类又分为监督分类和非监督分类。前者需要事先确定各个类别的标准及其训练区,并计算训练区像元灰度统计特征,然后将其他像元归并到相应的类别,常用的监督分类算法有最小距离法、波谱曲线匹配法、Fisher 线性判别法等;后者是一种未知类别标准

的分类方法,它直接根据像元灰度特征之间的相似性和相异性进行合并与区分,形成不同的类别,常用的非监督分类算法有聚类法和分裂法等。近年来,专家系统分类法和神经网络分类法等也已在遥感图像分类中获得较好的分类效果。

遥感信息的这一系列处理技术,自从计算机图像处理技术普及以来取得了长足的进步。处理技术的发展,极大地推进了遥感技术的应用。反之,遥感技术的发展和运用,又对卫星图像处理技术提出了更高的要求。在这种形势下,针对遥感图像信息的特点,可以将卫星图像处理技术分为预处理技术和应用处理技术两大类。预处理技术主要针对遥感数据获取的特色,进行一系列的校正,以消除引入的误差和干扰等,使遥感数据能较真实地反映它所代表的物体结构、光谱等方面的信息。应用处理技术则必须从研究对象的地学、生物学或环境科学等方面的综合知识入手,深入研究其波谱特性及其在探测和处理过程中的变化,结合遥感基础理论知识,并通过利用不同信息(如多平台、多时相遥感信息等)和技术(如 GIS、信息复合技术等)的优势,使得能从卫星遥感记录中获得研究对象的丰富内容,提取到能反映研究对象本质特征的专题信息。如何才能有效地进行这种处理,这是一个富有吸引力和具有潜力的研究方向。

参考文献

1. 陈述彭主编. 遥感大辞典. 北京: 科学出版社, 1990
2. 陈述彭主编. 地球系统科学. 北京: 中国科学技术出版社, 1998
3. 戴昌达, 姜小光, 唐伶俐. 遥感图像应用处理与分析. 北京: 清华大学出版社, 2004

(黄杏元)

yemian miaoshu yuyan

页面描述语言 (page description language)

排版软件和文字处理软件中对输出页面的内容和格式(如文字和图形的位置、形状、大小及颜色等)进行描述的一种语言。

页面描述语言是排版软件与输出设备(如激光印刷机、激光照排机等)的中间媒体,排版软件把排版结果输出成一种页面描述语言的形式,输出设备则解释这种由页面描述语言描述的页面并把结果输出到纸张、胶片等介质上。页面描述语言的典型例子有美国 Adobe 公司的 PostScript 和 PDF, HP 公司

的 PCL 等。

页面描述语言于 20 世纪 70 年代问世,至今已有 30 多年的历史。早期激光照排系统都设计了自己专用的、低级的页面描述语言,是页面描述语言的初级阶段。这个时期的页面描述语言描述能力很差,描述图形和图像的功能也很弱。

到了 80 年代,激光打印机的推广应用和激光照排系统的迅速发展,出版面的式样、色彩、质量等有了很大提高,对页面描述语言的要求提高了,特别是希望能提供一种不依赖输出设备硬件性能参数的页面描述语言。1985 年,Adobe 公司推出的 PostScript(Level 1) 成为页面描述语言发展史上的里程碑。

PostScript 是一种类似于 FORTH 语言的采用后缀表示法的解释性语言,具有很强的图像、图形和文字排版的描述能力。它既是面向光栅输出设备的页面描述语言,又内嵌了通用程序设计语言的许多特性,还提供了许多功能很强的内建的图形功能,也是用于控制打印机和显示器之类光栅输出设备的一种交互系统。

PostScript 推出后,首先在 Apple 公司的激光打印机上实现,此后一直被商用或办公用的高档打印机所采用。由于 PostScript 具有很强的图形功能,高质量的字模以及对页面描述的设备无关性,因此很快就被工业界普遍接受,成为事实上的工业标准。

90 年代,页面描述语言得到进一步发展。1990 年和 1997 年 Adobe 公司分别推出了 PostScript(level 2) 和 PostScript(level 3)。与第一版相比,它们对文字、图形、图像的描述功能和效率都有显著提高,进一步巩固了 PostScript 作为标准页面描述语言的地位。

国际标准化组织 ISO/IEC 在 PostScript(level 2) 的基础上,增加了文档生成属性等功能,制定了页面描述语言的国际标准(SPL)。但除了少数国家(如日本)使用之外,大多数国家并不支持 SPL 的应用。

使用 PostScript 语言描述的电子文档,称为 PostScript 文件,其文件扩展名为 .ps,现在已经成为计算机之间文件交换的标准格式之一。此外,Adobe 公司还开发了另一种电子文档的可移植表示格式(PDF)。PDF 和 PostScript 具有相同的成像模型,两种文件格式相互间可以直接进行转换,它们产生的打印输出的结果也完全相同。但 PDF 没有 Post-

Script 所具有的通用程序设计语言的功能。PDF 文档采用了静态的数据结构,适合有效地进行随机访问,所包含的导航信息允许用户交互地阅读 PDF 文档。

参考文献

Adobe Systems Incorporated. PostScript Language Reference Manual (Third Edition). 1999 (张福炎)

yijie luoji

一阶逻辑 (first order logic) 研究由个体、函数及关系构成的命题,由这些命题经使用量词和命题联结词构成的更复杂的命题以及各种命题之间的推理关系的逻辑。在一阶逻辑中,量词仅作用于个体变元。一阶逻辑是数理逻辑中发展得最为成熟的部分。在为数学的语言和推理建立形式系统的过程中,它处于核心地位,又称谓词逻辑。

传统逻辑主要是指古代希腊的亚里士多德逻辑,在中世纪被认为是金科玉律,完善无缺,不容许有任何更改。但是到了 19 世纪,人们觉得它有很多缺点,需要改革。传统逻辑仅限于讨论主宾式语句和三段论形式的推理,并且缺乏对于量词的研究。A. De Morgan 研究和发展的关系逻辑,提出了论域的概念。W. Hamilton 对量词进行了研究。G. Frege 于 1897 年建立了第一个谓词逻辑的形式系统。B. Russell 和 A. Whitehead 于 1910 年在他们的数学名著《数学原理》中总结了前一段的成果,建立了一个完全的谓词演算。1930 年 K. Godel 证明了谓词演算的完全性。

在命题逻辑中,不进一步分析原子命题的内部结构,原子命题被看作是是不可分割的最小单位,因而不能包括某些正确的推理。例如以下的推理:

每个有理数都是实数,
1 是有理数,
所以,1 是实数。

在这个推理中,前提和结论是三个不同的命题。在命题逻辑中,其推理形式只能表示成

$$\begin{array}{l} p \\ q \\ \hline r \end{array}$$

这不是命题逻辑中正确的推理形式。但这个推理是正确的,其正确性只有通过分析各命题中的个体词、谓词、量词才能揭示出来。这正是一阶逻辑所研究的。

在一阶逻辑中描述一个数学理论,首先会涉及

这个理论所讨论的对象、定义在这些对象上的函数以及这些对象之间的关系。数学理论所讨论的对象称为个体,由个体组成的非空集合称为论域或个体域。相等关系是经常需要用到的,在一阶逻辑中用一个特殊的符号,即等号“=”表示它。

为了表达每个个体都有某性质,在一阶逻辑中引进了全称量词 \forall 。为了表达至少有一个个体有某性质,在一阶逻辑中引进了存在量词 \exists 。例如,设论域是整数集, $\mathbf{N}(x)$ 表示 x 是自然数。 $\forall x\mathbf{N}(x)$ 表示命题“每个整数都是自然数”,这是一个假命题。 $\exists x\mathbf{N}(x)$ 表示命题“至少有一个整数是自然数”,这是一个真命题。

一阶逻辑使用的形式语言称为一阶语言,它的符号包括以下几类。

(1) 个体变元 x, y, z, \dots , 简称为变元。

(2) 函数符号 f, g, h, \dots ; 个体常元 a, b, c, \dots , 谓词符号 P, Q, R, \dots , 其中包括二元谓词符号“=”。

(3) 命题联结词 \neg, \rightarrow 和全称量词 \forall 。

通常称函数符号、个体常元、除等号“=”之外的谓词符号为非逻辑符号,而称其余符号为逻辑符号。不同的一阶语言有不同的非逻辑符号,而所有一阶语言的逻辑符号都是相同的。

因为 $\wedge, \vee, \leftrightarrow$ 可用 \neg, \rightarrow 定义,所以可取 \neg, \rightarrow 为基本联结词(参见命题逻辑)。因为存在量词 \exists 可用 \forall 和 \neg 定义, $\exists x\mathbf{N}(x)$ 与 $\neg\forall x\neg\mathbf{N}(x)$ 表达的命题之真假意义相同,所以仅取 \forall 为基本量词。

项的形成规则如下:

(1) 变元和个体常元是项;

(2) 若 f 是 n 元函数符号, t_1, \dots, t_n 是项,则 $f(t_1, \dots, t_n)$ 是项;

(3) 每个项都可以通过有穷次应用(1)和(2)获得。

公式的形成规则如下:

(1) 若 P 是 n 元谓词符号, t_1, \dots, t_n 是项,则 $P(t_1, \dots, t_n)$ 是公式,也称为原子公式;

(2) 若 A 是公式,则 $\neg A$ 是公式;

(3) 若 A, B 是公式,则 $(A \rightarrow B)$ 是公式;

(4) 若 A 是公式, x 是变元,则 $\forall x A$ 是公式;

(5) 每个公式都可以通过有穷次应用(1)~(4)获得。

如果变元 x 出现在公式 A 中形如 $\forall x B$ 的部分,则称 x 在 A 中的这次出现为约束出现,否则称为自由出现。例如,在公式 $P(x) \rightarrow \forall x Q(x)$ 中,第一个 x 是自由出现,后两个 x 是约束出现。如果变元 x 在

公式 A 中有自由出现,则称 x 为 A 的自由变元。没有自由变元的公式称为闭公式。常用 $A_x[t]$ 表示将公式 A 中 x 的所有自由出现代之以项 t 所得到的公式。如果 $A_x[t]$ 和 A 中的变元的约束出现数相同,则称 t 对 A 中的 x 是可代入的。例如 $f(z)$ 对于 $P(x) \rightarrow \forall y Q(x, y)$ 中的 x 是可代入的,而 $f(y)$ 对于 $P(x) \rightarrow \forall y Q(x, y)$ 中的 x 不是可代入的。

设 L 是一个一阶语言。可指定一个非空集合为论域,将等号“=”解释为论域上的相等关系,为 L 的每个个体常元指定论域中的一个个体,为 L 的每个 n 元函数符号指定论域上的一个 n 元函数,为 L 的每个非逻辑的 n 元谓词符号指定论域上的一个 n 元关系,就得到 L 的一个结构。闭公式在一个结构中的解释是一个命题。例如,指定论域为正整数集,个体常元 a 解释为2, $P(x)$ 解释为 x 是素数, $L(x, y)$ 解释为 $x \leq y$,则闭公式 $P(a) \wedge \forall y (P(y) \rightarrow L(a, y))$ 就解释为真命题:2是最小的素数。有自由变元的公式在一个结构中的解释还不是一个命题,因为自由变元值不确定。例如,在上面所给的结构中,如果给变元 z 赋值2,则公式 $P(z) \wedge \forall y (P(y) \rightarrow L(z, y))$ 解释为真命题;如果给变元 z 赋值3,则该公式解释为假命题。设给定一个结构 U ,如果给自由变元赋予论域中任何个体,公式 A 都被解释为真命题,则称 A 在 U 中有效,记为 $U \models A$ 。如果公式 A 在每个结构中有效,就称 A 为逻辑有效式或永真式,记为 $\models A$ 。例如,公式 $\forall x P(x) \rightarrow P(y)$ 是逻辑有效式,而 $P(y) \rightarrow \forall x P(x)$ 不是逻辑有效式。

谓词演算把逻辑有效式组成了一个完全形式化的公理系统。在谓词演算中,取某些逻辑有效式为公理,并规定了一些推理规则,以推导出所有的逻辑有效式。人们给出了许多等价的谓词演算系统。下面举出其中的一个。

取以下7种形式的公式为公理:

$$A \rightarrow (B \rightarrow A)$$

$$(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$$

$$(\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B)$$

$$x = x$$

$x = y \rightarrow (A \rightarrow A_x[y])$, 其中 y 对于 A 中的 x 可代入

$$\forall x A \rightarrow A_x[t], \text{ 其中项 } t \text{ 对于 } A \text{ 中的 } x \text{ 可代入}$$

$\forall x (A \rightarrow B) \rightarrow (A \rightarrow \forall x B)$, 其中 x 不是 A 的自由变元

推理规则有以下两条:

分离规则 由前提 A 和 $A \rightarrow B$ 推出结论 B 。

概括规则 由前提 A 推出结论 $\forall xA$ 。

谓词演算的定理是这样定义的:

- (1) 每个公理都是定理;
- (2) 如果 A 和 $A \rightarrow B$ 是定理, 则 B 是定理;
- (3) 如果 A 是定理, 则 $\forall x A$ 是定理;
- (4) 每个定理都可以通过有穷次, 应用(1)~(3)获得。

若公式 A 是定理, 则记为 $\vdash A$ 。

谓词演算的公理都是逻辑有效式。推理规则保证, 当前提都在某结构中有效时, 结论也在该结构中有效。因此, 谓词演算的定理都是逻辑有效式。这个性质称为谓词演算的可靠性。反之, 每个逻辑有效式都是谓词演算的定理。这是谓词演算的完全性, 由 K. Godel 于 1930 年证明。

公式是不是逻辑有效式是半可判定的, 不是可判定的(参见可计算性理论)。

可以用一阶逻辑刻画数学理论, 常把谓词演算的公理称为逻辑公理。除了逻辑公理之外, 数学理论还需要非逻辑公理, 它们刻画了该数学理论的研究对象的共同性质。这样的数学理论称之为—阶理论。如果—阶理论 T 的每个非逻辑公理都在结构 U 中有效, 就称 U 是 T 的模型。下面举出几个常见的一阶理论。

全序理论只有惟一的非逻辑符号, 即二元谓词符号 \leq 。它有 4 个非逻辑公理:

$$\begin{aligned} x &\leq x \\ (x \leq y \wedge y \leq x) &\rightarrow x = y \\ (x \leq y \wedge y \leq z) &\rightarrow x \leq z \\ x &\leq y \vee y \leq x \end{aligned}$$

每个全序结构都是它的模型。

群论有两个非逻辑符号: 个体常元 e 和二元函数符号“ \cdot ”。它有 3 个非逻辑公理:

$$\begin{aligned} \forall x \forall y \forall z ((x \cdot y) \cdot z &= x \cdot (y \cdot z)) \\ \forall x (x \cdot e &= x \wedge e \cdot x = x) \\ \forall x \exists y (x \cdot y &= e \wedge y \cdot x = e) \end{aligned}$$

每个群都是它的模型。

初等数论的非逻辑符号有: 个体常元 0, 一元函数符号 S , 两个二元函数符号“ $+$ ”和“ \cdot ”。它的非逻辑公理为:

$$\begin{aligned} \forall x \neg (S(x) &= 0) \\ \forall x \forall y (S(x) &= S(y) \rightarrow x = y) \\ \forall x (x + 0 &= x) \\ \forall x \forall y (x + S(y) &= S(x + y)) \\ \forall x (x \cdot 0 &= 0) \end{aligned}$$

$$\forall x \forall y (x \cdot S(y) = (x \cdot y) + x)$$

$$A_x[0] \wedge (\forall x (A \rightarrow A_x[S(x)])) \rightarrow \forall x A$$

自然数系统是它的一个模型。

—阶逻辑有很强的表达能力, 用—阶理论能够刻画许多数学结构。但是, —阶逻辑不是万能的, 其表达能力受到一定的限制。

如果—阶理论 T 的模型的类恰好是结构的类 S , 就称 T 刻画 S 。例如, 全序理论刻画全序结构的类, 群论刻画群的类。常把论域是有穷集的结构称为有穷结构。

—阶逻辑有可靠且完全的公理系统, 有许多良好的性质, 是数理逻辑中发展得最为成熟的部分。在计算机科学和人工智能的各个领域, 如数据结构理论、程序设计语言的形式语义, 程序正确性证明、软件规范、程序变换、逻辑程序设计、定理的机器证明、常识表示和推理等方面, 都广泛地应用了—阶逻辑的概念、理论和方法。

参考文献

1. 王宪钧. 数理逻辑引论. 北京: 北京大学出版社, 1982
2. 王浩. 数理逻辑通俗讲话. 北京: 科学出版社, 1981
3. Kleene S C 著. 元数学导论. 莫绍揆译. 北京: 科学出版社, 1984 (何自强)

yidongshi jisuanji

移动式计算机 (mobile computer) 在移动过程中可以进行信息处理的、可随身携带的、体积很小的微型计算机; 又曾称为便携式计算机。

移动式计算机可分为**笔记本计算机**、**手持计算机**和**可穿戴计算机**(参见**可穿戴计算**)等。这些移动式计算机都能通过无线或有线传输方式与通信网络连接以传输和处理信息, 它们具有成为因特网或其他通信网络终端设备的功能, 有的还具有电话和传真等功能, 如**个人数字助理 (PDA)**、现代化的移动手持电话机、**全球定位系统 (GPS)**终端等。它们都具有能处理业务信息、个人信息、上网浏览和进行电子邮件通信的功能。另外, 它们还具有节能的共同特点, 即采用节能的低电源电压的、互补金属氧化物半导体 (CMOS) 工艺的芯片, 以及节能的外围设备, 如代替硬磁盘存储器的闪存和液晶显示屏。在输入方面, 常采用笔输入技术, 即使用模式识别技术, 以笔输入方式代替键盘输入。

移动式计算机发展很快, 为了便于大规模生产,

建立了很多标准。例如,为了使移动式计算机与电话结合,建立了电话应用编程接口(TAPI);为了使移动式计算机板卡的可互换性更高,建立了可互换板卡体系结构(ExCA);特别是为了方便地扩展存储音量和外围设备,采用了由个人计算机存储卡国际协会(PCMCIA)所制定的标准。移动式计算机可选的电路板卡种类繁多,例如存储卡、调制解调卡、以太网卡、传真卡等。移动式计算机的外围设备也日趋多样化,如跟踪球、触屏、控制杆等。为了使移动式计算机有硬拷贝输出,也出现了一些小型便携式输出设备,如热转印印刷机(参见非击打式印刷机)等。

移动式计算机的操作系统一般采用嵌入式操作系统。嵌入式操作系统的特点是代码十分紧凑,存储总容量在几万到儿兆字节,以节省电源消耗量。嵌入式操作系统往往具有基本功能的核,然后根据应用的不同或系列产品的升级,再在核外面扩充操作系统功能。很多嵌入式操作系统是针对专门用途而开发的,有的还要求有实时的功能。移动式计算机中的处理机的功能范围变化较大,可以用只有几万个晶体管的处理机,也可用多达几千万晶体管的复杂的处理机,如 Pentium 4 低功耗处理机。

移动式计算机的用途广泛,随着有线和无线网络技术的迅速发展,将有更广阔的应用前景。人们可携带笔记本电脑或 PDA 外出办公;可在移动过程中用 PDA 或手机上网用电子邮件和其他人通信;当移动式计算机用于汽车中时,可以显示汽车当前的位置和行驶路径的地图;当和全球定位系统(GPS)相结合时,可使野外人员知道自己所在地精确位置与周围情况,如果移动计算机能有语音输入与输出功能,则可和其他人通话。(李三立)

yidong shujuku

移动数据库(mobile database) 支持移动计算环境的数据库。它能满足人们在任意地点、任意时间访问任意数据的需要。移动计算环境指为移动式计算机和无线传输的应用而创建的网络运行环境。移动数据库涉及数据库技术、分布式计算以及移动通信等多个学科领域。

移动计算环境的特殊性给移动数据库的研究带来了新的挑战。在移动数据库中需要考虑诸多传统计算环境下无需考虑的问题,如对移动性及位置相关查询的支持、对断接操作的支持、对跨区长事务的支持、对查询优化的特殊考虑、对提高有限资源的利

用率及系统效率的考虑等。为了有效地解决上述问题,如下关键技术 in 移动数据库中具有特别的意义:移动数据库复制技术、移动事务处理技术、移动对象数据库技术、位置相关数据的处理技术、位置相关的查询处理、数据广播及移动信息发布等。

数据复制是提高分布式处理系统可用性和可靠性的一项关键技术。为了提高移动数据库系统的性能,人们对移动数据库复制技术进行了大量研究,并取得不少成果,如两级复制算法、虚拟主复本方法、多版本冲突消解方法和三级复制体系结构等。

在移动数据库中,固定主机和移动计算机均能发起事务(参见事务元),其中移动计算机发起的事务称为移动事务。由于有限的通信带宽以及频繁断接操作的影响,移动事务通常属于长事务;在移动事务执行过程中,移动计算机位置的改变会带来复杂的过区切换问题;此外,移动事务执行时更容易出错,且要访问更加复杂的数据资源。移动事务的上述特点使得移动事务处理成为一个具有挑战性的研究领域,其中移动性和长事务特性是移动事务模型需要着重解决的问题。人们已经提出了许多移动事务处理模型。其中有些模型,如集群事务模型、Kangroo 事务模型、自适应移动事务模型、MOFLEX 事务模型等,对事务的不可再分割性、一致性、隔离性与持久性(ACID)做了一些折中或者重新定义,因此不能完全保证事务的可串行性和数据库的一致性。尽管对于许多移动数据库应用而言,将可串行性作为正确性的标准可能过于苛刻,但许多重要的应用仍然需要用可串行性作为标准来保证数据库的一致性。

移动对象数据库(MOD)是指对移动对象的位置及其他相关信息进行表示与管理的数据库。在移动对象数据库中通常管理着大量的移动对象,这些移动对象的位置是不断变化的,如汽车、飞机及其他移动用户等。越来越多的应用要求对移动对象进行管理,而定位技术和无线通信技术的发展使得跟踪和记录移动对象的位置成为可能。

参考文献

1. Imielinski T and Badrinath B R. Data Management for Mobile Computing. SIGMOD RECORD, 1993, 22(1): 349
2. Imielinski T, Badrinath B. Mobile wireless computing: Challenges in data management. Comm. of the ACM 1994, 37(10): 18 ~ 28
3. Wolfson O, Xu B, Chamberlain S, Jiang L.

Moving Object Databases: Issues and Solutions. In: Proceedings of the 10th International Conference on Science and Statistical Database Management, pp. 111 ~ 122. Capri, Italy, July 1998 (孟小峰)

yidong tongxin

移动通信 (mobile communication) 见移动通信网。

yidong tongxin wang

移动通信网 (Mobile Communication Network) 参与通信的一方或双方可以在其作用范围内随意移动并进行通信的网络,而且在参与通信的双方中至少有一方处于运动中或暂时停留在某一非预定的位置上。移动通信网按使用对象、用途、经

营方式、频段、制式、入网方式等有不同分类方法。如按使用对象分,可分军用、民用;按用途和区域分,可分陆上、海上、空间;按经营方式分,可分为公众网、专用网等。

典型的移动通信网组成方案如图1所示。它由移动通信交换局(MTX)、基站(BS)、移动台(MS)和局间或局站间的中继线组成。移动台和基站、移动台和移动台之间采用无线传输方式。基站与移动通信交换局,移动通信交换局与有线网(PSTN)之间一般采用有线(中继线)方式进行信息传输。移动交换局和基站担负信息交换和接续以及对无线频道的控制等。基站与移动台都设有收发信机,收发信共用装置和天线、馈线等。每个基站都有一个有发信功率与天线高度所限定的地理覆盖范围,称为覆盖区。由多个覆盖区组成全系统的服务区。

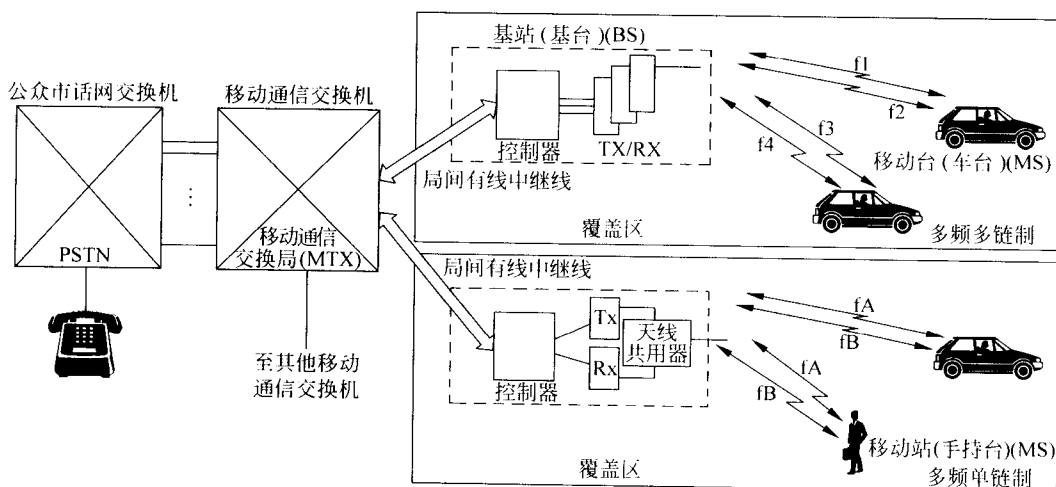


图1 移动通信网的组成

移动通信网与有线固定网相比,具有下列特点:
①采用复杂的无线电波传播模式;②干扰多而复杂;③工作环境恶劣,设备要求可靠性高,体积小、重要轻和省电等;④主要通过有效利用频率技术扩大用户容量;⑤组网方式多样,网络结构和信令方式比较复杂。

移动通信的基本技术主要包括:①调制技术;②移动信道中电波传播的模式及抗衰落措施;③抗干扰措施;④组网技术;⑤移动台的小型化。

参考文献

1. 田翠云,赵荣黎,蒋忠涌. 移动通信系统. 北京:人民邮电出版社,1990

2. 郭梯云,郭国扬,张厥盛. 移动通信. 西安:西安电子科技大学出版社,1995 (史美林 英春)

yichuan suanfa

遗传算法 (genetic algorithm) 用计算机模拟生物演化的自然选择过程来求解复杂问题的一类计算方法。一般认为它是由 J. H. Holland 提出的^[1],他的原意与其说是求解特别的问题,不如说是试图学习自然系统对环境变化的自适应过程来设计一个具有类似属性的人工系统。这种思想可以追溯到 A. M. Turing 1948 年的文章:智能机器 (Intelligent machines)。他认为达到机器智能的方法就是进行遗

传或演化搜索。在 1950 年的论文: 计算机与智能 (Computing Machinery and Intelligence) 中他描述了怎样用演化与自然选择来自动创造智能机器: “我们不能期望在第一步尝试中就能找到一个好的机器。人们试着去教导这样一种机器并看它学习得怎样, 继而去训练另一个, 看它是好些还是坏些。只要做一比较就知道这种培训过程与演化之间存在着一种明显的联系:

机器的结构 = 遗传物质

机器的变化 = 变异

试验者的评判 = 自然选择。”

Holland 把 Turing 这种构思精确化为一种计算方法, 即遗传算法。

下面以求解优化问题为例来描述 Holland 的遗传算法。

求 $x^* \in S$, 使得

$$f(x^*) \geq f(x), \quad \forall x \in S$$

其中 S 为问题的解空间, $f: S \rightarrow R$ 为适应函数。

遗传算法:

Begin

$t := 0$;

initailize $P(t) := \{x_1(t), x_2(t), \dots, x_n(t)\}, x_i \in S$;

evaluate $P(t); f(P(t)) = \{f(X_1(t)), f(X_2(t)), \dots, f(X_n(t))\}$

$P(t) := \text{selection}[P(t)]$;

while not terminate do

$P_c(t) := \text{crossover}[P(t)]$;

$P_m(t) := \text{mutation}[P_c(t)]$;

$P(t+1) := \text{selection}[P(t) \cup P_m(t)]$;

$t := t + 1$;

endwhile

end

其中 t 为演化的代的数, $P(t)$ 为第 t 代群体, $x_i(t)$ 为其个体, 群体的规模为 n 。这一演化过程的动力学行为可以形式化地描述为一阶随机差分方程:

$$P(t+1) := \text{selection}[\text{mutation}[\text{crossover}[P(t)]]]$$

遗传算法包括下述六个主要组分:

(1) 解的表示 个体 x 的染色体编码 $E: S \rightarrow G$, 它是表型空间 S 向基因型空间 G 的映射。

(2) 解的评价 适应函数 $f(x)$ 评价解的好坏, 它是驱动演化过程的动力。

(3) 遗传算子 杂交算子与变异算子等, 用来

改变染色体的结构, 实现基因空间中的搜索。

(4) 淘汰法则 赌轮法则、排名法则、锦标赛法则等, 用来实现达尔文的优胜劣汰原理。

(5) 参数设置 群体规模 n , 杂交概率 p_c , 变异概率 p_m 和淘汰压力等。

(6) 终止条件 演化的最大代数或解所达到的精确度等。

遗传算法的设计主要就围绕这六个方面进行。以一个 8 位长度的位匹配问题为例来说明算法的具体实现过程:

1. 解的表示: 个体 x 为一个长度为 8 的二进制位串。其表现型空间 S 和基因型空间 G 相同: $S = G = \{00000000, 00000001, \dots, 11111111\}$ 。

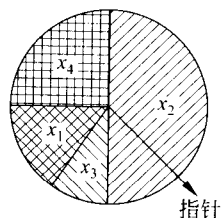
2. 解的评价: 适应函数 $f(x)$ 为个体 x 中 1 的个数, 如 $f(10011011) = 5$ 。

3. 参数设置: 群体规模 $n = 4$, 杂交概率 $p_c = 0.7$, 变异概率 $p_m = 0.001$ 。

若随机初始化群体如下:

个体 $x(0)$	基因型	适应值 $f(x(0))$
x_1	00000110	2
x_2	11101110	6
x_3	00100000	1
x_4	00110100	3

4. 淘汰法则: 赌轮法则。根据个体适应值比较(个体 x_1 比例为 $2/(2+6+1+3) = 2/12$, x_2 为 $6/12$, x_3 为 $1/12$, x_4 为 $3/12$) 构造如下图所示的赌轮:

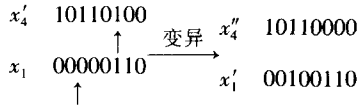


转动赌轮指针 4 次, 选择 4 个个体分别为 x_2, x_4, x_2 和 x_1 。构成 $P(0) = \{x_2, x_2, x_4, x_1\}$ (这时 x_3 已被淘汰出局)。

5. 遗传算子: 采用一个杂交算子与一个翻转变异算子($0 \rightarrow 1, 1 \rightarrow 0$)。根据杂交概率 p_c , 假设选取 x_2 与 x_4 为父体杂交(交换子串, 如下图所示)产生后代 x'_2 和 x'_4 , 其余个体不变, 则

x_2	11101110	杂交	x'_2	01101110
x_4	00110100		x'_4	10110100

杂交操作后群体变为 $P_c = \{x_1, x_2, x'_2, x'_4\}$ 。根据变异概率 p_m 对每个个体进行变异操作,假设 x'_4 和 x_1 变异(随机变换一位,如下图所示)为新个体 x''_4 和 x'_1 , P_c 的其余个体不变,



则经过变异后,形成群体 $p_m = \{x'_2, x''_4, x_2, x'_1\}$ 并评价如下:

个体 $x(1)$	基因型	适应值 $f(x(1))$
x_1	01101110	5
不变 x_2	10110000	3
x_3	11101110	6
x_4	00100110	3

6. 停机条件:一代一代地重复上述演化过程,直到达到最大适应值 $f(x) = 8$,即找到最优个体 $x_{opt} = 11111111$ 时才停机。

由于遗传算法模拟了生物的演化过程,故它具有某些“智能”特征:自适应性、自组织性、自学习性与自优化性,即

自适应性:以代表环境的适应函数为驱动力使群体朝着一定的目标演化;

自组织性:群体通过杂交、变异,不断地改变与调整自身的结构,形成新的群体;

自学习与自优化性:通过竞争与淘汰,吐故纳新,不断“学习”,不断优化。

与其他算法相比,遗传算法具有下述特点:

原理的简明性,故易学易用;

演化的随机性,故搜索能力更强,算法更鲁棒;

搜索的群体性,故具有全局优化的能力;

计算的并行性,故可充分发挥新一代并行计算机系统的性能;

应用的广泛性,可应用到不同的学科领域,特别是用来求解传统算法不易解决的高难度计算问题。

半个世纪以来,已提出用计算机模拟大自然的演化过程,特别是生物的演化过程来求解复杂问题的各式各样的计算模型。在仿生演化方面,除遗传算法外,还有 1963 年由 I. Rechenberg 与 H-P. Schwefel 提出的演化策略(ES);1966 年由 L. J. Fogel, A. J. Owens 与 M. J. Walsh 提出的演化规划(EP)以及 1992 年由 J. Koza 扩展遗传算法而提出的遗传程序设计(GP)。在模拟物理系统的演化过程方面,有 1983 年由 S. Kirkpatrick, C. D. Gelatt Jr. 与

M. P. Vecchi 提出的模拟退火算法(SA);1980 年由康立山、陈毓屏提出的弹性混乱松弛法以及 1987 年由 R. Durbin 与 D. Willshaw 提出的弹性网方法(EN)。由于他们所研究的领域与背景不同,所面临的问题各异,所采用的计算工具的差别,特别是大自然演化过程的复杂性与多样性,虽然有着相同的灵感来源,但其设计的计算模型却有着各自的特色。这些计算模型在发展与应用过程中互相学习,互相渗透,到 1992 年就逐渐融合为一个难分彼此的统一计算模型:演化计算。此后,在模拟社会系统演化过程方面,有陈火旺、吴少岩、张青富提出的家族优生进化算法,由鄢烈祥提出的列队竞争算法,由黄文奇、金人超提出的拟人算法,由查凯、曾建潮、孙承毅提出的思维进化算法,由 M. Dorigo, V. Mamiezzo 和 A. Colomni 提出的蚁族系统(ACS)以及由 R. G. Reynolds 提出的文化算法(CA)。还有模拟神经系统、免疫系统、生态系统以及其他生物系统的许多演化算法。

演化算法可以描述如下:

演化算法:

Begin

$t := 0$;

initialize $P(t)$;

evaluate $P(t)$;

while not terminate do

$P'(t) := \text{variation}[P(t)]$;

evaluate $[P'(t)]$;

$P(t+1) := \text{selection}[P'(t) \cup Q]$;

$t := t + 1$;

endwhile

end

其中 $P(t)$ 为第 t 代群体, Q 为参与第 t 代竞争的其他个体组成的群体。例如取 $Q = P(t)$,或是历代保留下来的精英个体的集合,它也可以是空集,即父代不参与子代的竞争。variation 在仿生演化算法中为遗传操作,如变异、杂交或重组等。在拟物演化算法中可以是任何种类的扰动法则,如梯度方向法、弹性松弛法、随机游动法等。Selection 在仿生演化中是达尔文的优胜劣汰原理,在拟物演化算法中它可以是上山法,也可以是 Monte-Carlo 或 Metropolis 式的随机决策法。总之,算法的具体细节是与所模拟的系统的演化过程相关的。更可采用仿生、拟物与拟人相结合的混合演化算法。

演化算法可以形式化地描述为一阶随机差分

方程:

$$p(t+1) := \text{selection}[\text{variation}[P(t)]], \quad t=0, 1, 2, \dots$$

其动力学行为,包括它的收敛性、收敛速率以及计算复杂性等,是演化计算理论工作者探索的热点;而随机算子 selection · variation 的最优设计与实现,则是演化计算应用工作者追逐的目标,特别是它的并行实现。

遗传算法除了作为演化计算的杰出代表而备受关注外,还在于它应用的广范性。这些应用包括函数优化、组合优化、自动程序设计、机器学习、网络设计、金融预测等不胜枚举。由于演化计算原理的简明性、演化的随机性、搜索的群体性、计算的并行性与应用的广泛性,已使它渗透到了许多学科领域。又由于它的自适应性、自组织性、自学习性与自优化性等“智能”特征,故每当它深入结合到某一个学科领域时就会引起该学科的巨大变化,甚至形成新的学科领域,如演化机器人学、演化软件、演化硬件、演化电子学、演化密码学、演化经济学、演化建模学、演化设计学、演化通信学、DNA 与分子计算机、量子计算机等。

演化计算为什么要模拟大自然的演化过程呢?因为大自然是一位最伟大的设计师,它设计了万事万物,包括万物之灵的人类。学习它的设计技术——演化过程,就抓住了问题的本质。

参考文献

1. Holland J H. Adaptation in Natural and Artificial Systems. Ann Arbor, MI: University of Michigan press, 1975 (Second edition: MIT press, 1992.)
2. Turing A M. Intelligent machines. Papers 21~23, in Mechanical Intelligence: Collected Works of A. M. Turing, D. C. Ince (ed.). North-Holland: Amsterdam, 1992, 107~128
3. Michalewicz Z, Fogel D B 著. 如何求解问题——现代启发式方法. 曹宏庆,李艳,董红斌,吴志健译. 北京:中国水利水电出版社,2003 (陆玉昌)

yichuan xuexi

遗传学习 (genetic learning) 基于物种变异和自然选择的原理建立的一种机器学习方法。

遗传学习将搜索结构编码为字符串形式,每个字符串结构称为个体。然后对一个集合的字符串结构(群体)进行循环操作,每次循环称为一代。其中,字符串模拟染色体的作用,单个位模拟基因的作用。根据评价函数,每个个体给予一个数值估算,称为遗传适应度。选择适应度高的个体参加操作,不

断迭代,使之优者生存,劣者淘汰,从而达到不断进化,可得到有高适应度的群体。

在将遗传原理应用于人工系统的搜索之前,已有一些生物学家用计算机仿真遗传系统。虽然这些研究的目的在于了解自然现象,但有的工作与现代遗传算法的概念很相近。1960年,A. S. Fraser 在生物仿真中,用(-1,1)间变动的表型值(函数值)选择基因型(字符串)作为父母,仿真后代字符串的变化。1962年,J. H. Holland 关于自适应系统的论文奠定了自适应系统的理论基础。他认为自适应系统的研究涉及到自适应系统和它的环境之间的相互作用,应该研究系统怎样有效地调节适应它们的环境的过程。J. D. Bagley 于1967年在他的博士论文中第一次提出遗传算法这个概念。他用遗传算法研究自适应游戏问题。同年,R. S. Rosenberg 用了 Holland 的模式理论研究了生物细胞仿真的问题。他着重研究用子孙代函数反映选择过程的竞争。1975年是遗传算法取得很大进展的一年。Holland 发表了重要著作“Adaptation in Natural and Artificial Systems(自然和人工系统中的适应性)”,系统地论述了遗传算法的基本理论,为遗传算法的发展奠定了基础。同年,K. A. Dejong 把 Holland 的模式理论和他自己的计算实验结合起来,研究数据结构设计、算法设计、计算机操作系统自适应控制等,得到一些很有意义的结论和方法,为遗传算法和它的应用建立了坚实的基础。经过长期研究,D. E. Goldberg 成功地将遗传算法应用在搜索、优化和机器学习中。1989年他发表了著作“Genetic Algorithms”,清楚地阐明遗传算法的实现和应用。80年代以来,遗传学习的理论和应用得到了迅速的发展,已被广泛应用到工程、生物学和社会科学中。

遗传学习的基本思想来源于自然进化过程。在自然进化过程中,物种在复杂变化的环境中寻找最佳组合。物种得到的知识保存在染色体的构造中。物种繁殖时,染色体的构造会发生变化。修改染色体构造的基本操作有交换、突变和倒位等。交换是双亲的染色体间交换构造信息,使双亲中较好的基因在后代中组合起来。突变将在染色体中加入有益的结构。倒位改变染色体里基因的位置,使基因能聚集成簇,增加在遗传过程中一起移动的概率。

遗传算法模拟生物的遗传过程,它的典型执行过程是:

- (1) 随机产生初始群体。
- (2) 当不满足算法终止条件时,重复执行下列步骤:

第一步,计算群体中每个个体字符串的适应度;

第二步,从当前群体选择产生新的群体;

第三步,对新的群体中的个体进行遗传重组操作(例如交换,突变等)。

(3) 在后代中最高适应度的个体字符串被指定为遗传算法运行的结果。

遗传算法包括六个基本部分:

(1) 问题解的染色体表示,即字符串表示模式。

(2) 产生初始群体的方法。

(3) 评价函数,决定染色体的优劣。

(4) 遗传操作,用以修改字符串的结构。

(5) 算法终止条件,确定算法运行的目标。

(6) 算法参数设置,包括群体大小、使用各种遗传操作的概率等。

一般,表示模式用 $\{0,1,\#\}$ 集上的位串方式,这里 $\#$ 为无关字符,它描述了字符串间的相似性。在遗传算法中有一基本的定理,称为模式定理。模式定理说明短的、低阶、性能高于平均值的模式在连续各代中被试验的次数呈指数规律增加。遗传操作不断产生出新的模式进行试验,使群体中表示的模式逐渐得到各自适当的试验次数,结果是适应度高的模式被存储在一个精选的相当小的群体中。这种短的、低阶、高适应度的模式称为构造块。遗传算法假定构造块能够组合起来形成较好的字符串,这种假定称为构造块假设。

自从 80 年代以来,遗传学习在学习产生式规则中得到广泛应用。遗传学习一般以基于特征的方法表示实例和事件,以多组模式的析取来表示获取的知识,其中每个模式规定了某些特征的存在与否。模式有一个称作适应度的权值,反映过去经验的执行性能。当遇到新实例时,系统便依据适应度强的模式给出结论。遗传学习的研究一般集中在监督学习和渐进式学习方式。遗传学习的执行任务一般用于分类和问题求解,尤其适用于非常复杂困难的环境,如带有大量噪声与不断变化的事件,问题目标不能精确明显地定义以及需要通过很长的运行过程才能确定当前行为的好坏等情形。

遗传学习在处理新的实例时一般有三个步骤:更新模式的适应强度;应用遗传操作;用适应度强的模式取代或淘汰弱的模式,以保证整个模式集(群体)的大小不变。

遗传学习典型地用于分类器系统以发现新规则。分类器系统是一种高度并行的、消息传递的、基于规则的系统,由三部分组成,即规则和消息部分、信任赋值部分以及遗传算法。其中遗传算法主要用

于从已有的适应度强的规则(由信任赋值部分给出)中生成新的候选规则。

遗传学习具有通用、简易和有效的特点,特别是在复杂空间能进行鲁棒搜索。遗传学习已经在大规模集成电路的电路层布局、输气管道系统的状态优化、旅行商问题、键盘配置设计、任务调度规划、通信网络连接优化、仿真单细胞机体的群体的进化、聚类算法、搜索游戏评价函数、自适应文本聚类等诸多方面得到了成功的应用。随着遗传学习被应用于解决越来越多的问题,遗传学习的基本理论有必要进一步完善,如深入理解遗传算法的本质特性和自适应原理;对遗传算法进行定量的研究;如何设置算法参数,预测算法执行的结果等。遗传学习内在的并行性使它能以并行方式高效地搜索复杂的问题空间。因此,在大规模并行计算机系统上如何实现遗传学习也是当前感兴趣的研究课题。将遗传学习和人工神经网络结合起来,产生新的问题求解方法,不断拓宽应用领域。

参考文献

1. Goldberg D E. Genetic algorithm in search, optimization, and machine learning. MA: Addison-Wesley Publishing Company INC, 1989
2. Shi Zhongzhi. Principles of machine learning. Beijing: International Academic Publishers, 1992

(史忠植 莫纯欢)

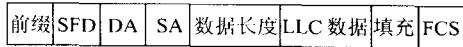
yitaiwang

以太网(Ethernet) 采用带碰撞检测的载波侦听多址访问(CSMA/CD)方法进行介质访问控制的一种局域网。1976年Xerox公司在1 km长的同轴电缆上建立了一个2.94 Mb/s、采用CSMA/CD方法,连接超过100个连接设备,取名为“Ether”的局域网。此后,得到DEC公司和Intel公司的支持,共同制定标准和研制器件,并公布了1.0版本(10 Mb/s)的以太网规范。这个规范后被IEEE标准委员会采纳,作了一些修改后成为IEEE 802.3标准(参见局域网协议标准)。

以太网结构中采用具有分接头的同轴电缆作为传输介质,将各站点互连。信息的传输速率为10 Mb/s。总线分布距离在0.5 km左右。当通信距离较大时,允许使用中继器进行分段连接。总线上的信号采用曼彻斯特编码(参见数据解码)。以太网采用格式固定的可变长数据帧形式传送。数据帧格式如图1所示。它由六个字段组成。网上的站点发送数据时,是以数据帧的形式发送到介质上。该帧

沿着介质传播到各个站点,只有那些地址与数据帧目的地址相符的站点才把它接收下来,这样就完成了一次数据传输。一个站要发送帧时,首先需监听总线以决定介质上是否有其他站的发送信号存在。如果介质空闲,则可以发送;如果介质忙,则等待一定间隔后发送。这就是载波监听多址访问(CSMA)。由于通道延迟,采用CSMA算法,当总线上两个站点监听到介质上没有信号存在而发送帧时,仍会产生冲突。这样,即使冲突发生,已被破坏的帧仍将发送完,使总线利用率降低。一种改进方案称作载波监听多址访问-冲突检测(CSMA/CD)被提出。每个站在发送帧期间,同时有检测冲突的能力。一旦检测到冲突,就立即停止发送,并向总线发一个阻塞信号,通知总线冲突已发生,冲突方要等待一个随机时间,然后再用CSMA算法发送。因此,CSMA/CD方案可以提高总线利用率。

字节数 7 1 2/6 2/6 2 0~1500 0~46 4

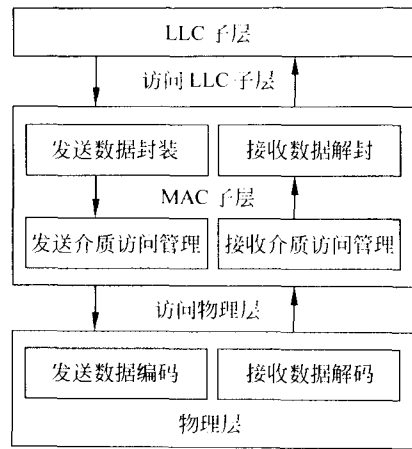


SFD=起始定界符; DA=目的地址; SA=源地址;
FCS=帧检验序列

图1 以太网数据帧格式

以太网遵循的IEEE 802.3标准对**局域网逻辑链路控制子层(LLC子层)**、**局域网介质访问控制子层(MAC子层)**和物理层的功能作了规定。这些层严格对应于**开放系统互连基准(参考)模型**的最低两层,而从LLC子层和MAC子层在一起完成数据链路层功能。各层通过接口相互作用,同时提供服务规范中的所规定的服务。

介质访问控制(MAC)子层的功能如图2所示。发送和接收介质访问管理模块的功能主要是实现CSMA/CD介质访问控制协议。当用户层要求发送数据时,先由链路层把数据按帧格式进行封装,然后



LLC——逻辑链路控制
MAC——介质访问控制

图2 MAC子层的功能

准备发送。链路发送管理通过随时检查由物理层提供的“载波侦听”信号了解传输介质的“忙”和“闲”。当检测到介质空闲时,在延迟一段时间后就开始发送。链路层向物理层发出串行数据流。由曼彻斯特编码器将时钟和数据转换成自同步的串行数据流。在发送期间有可能有多个站点同时侦听到介质空闲,并开始发送,从而导致碰撞。所以每一站点的物理层都设有碰撞检测机构,随时监视介质。若有碰撞,则置位“碰撞检测”信号,且继续发送称为阻塞码以强化碰撞,引起其他站点的注意,之后终止此次发送。经过一个伪随机延迟后,再重新发送。如果重发15次后仍不能无碰撞地发完,则表明总线过于繁忙,应放弃发送,并作为一个错误,报告给用户层。

以太网的物理介质规范给用户提供了方便的多项选择,如表1所示。

表1 以太网物理介质规范的参数

	10Base5	10Base2	10Base-T	10Broad36	10Base-F
传输介质	同轴电缆 (50Ω)	同轴电缆 (50Ω)	无屏蔽双 绞线	同轴电缆 (75Ω)	光纤 (850 nm)
电缆直径(mm)	10	5	0.4~0.6	0.4~1.0	62.5/125 μm
编码技术	曼彻斯特	曼彻斯特	曼彻斯特	DPSK	曼彻斯特
拓扑结构	总线	总线	星状	总线/树状	星状
最大段长度(m)	500	185	100	1 800	500
每段上的结点数	100	30	1 024		32
介质通信方式	基带	基带	基带	宽带	

当越来越多的站点加到以太网上时,会使得网络上的传输量急增,结果造成网络饱和。为解决这个问题,一种交换式以太网应运而生。这个系统有一个高速交换机,若干站点通过连接器与交换机互连。当一个站头要发送一个帧,首先检验目的站点是否在同一个交换机内。若是,则利用该高速交换机进行通信;否则通过交换机上的总线进行通信。

参考文献

1. Andrew S. Tanenbaum. Computer Networks, 3/e, Prentice Hall, 1996

2. 胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002 (徐伟氏)

yibu chuanshu

异步传输(asynchronous transmission) 不用共同时钟的定时机制来同步发送方与接收方之间事件的传输方式。它不是通过固定的时间间隔来分隔字符,而以字符流方式传送数据。

数字数据通信(模拟信号或数字信号)的基本要求是,接收方必须知道它所接收数据的每一位的开始时间和持续时间。满足这一要求的最早和最简单的方法是采用异步传输。这种方法一次传输一个字符(5~8位)的数据,每个字符用一个起始位引导,用一个停止位结束(如图1所示)。起始位的编码为0,有一位时间宽度。换句话说,起始位是具有0值的一位。停止位的值为1,而且有一个取决于系统的1~2位宽度的最小持续周期。如果没有发送的数据,那么发送方就发送连续的停止位。接收方根据从1~0的跳变来识别一个新字符的开始。为了把字符的所有位都恢复原状,接收方必须相当精确地把握每一位的宽度。但是,由于接收方用每个停止位重新同步,因此少量的漂移(例如每位1%的漂移)是无关紧要的。这种通信方法是简单的和便宜的,但是每个字符有2~3位的额外开销。因为每个字符是相互独立发送的,为了防止发送方和接收方的计时漂移,它们的时钟必须设法同步。一种方法是在发送方和接收方之间提供一条单独的时钟线,否则,就必须把时钟信息放入数据信号之中。对

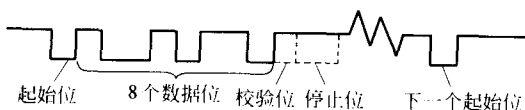


图1 异步传输

于数字信号,可以用曼彻斯特编码完成这种功能;对于模拟信号,可以使用许多技术来实现。载波频率本身可以根据载波的相位来与接收方同步。

参考文献

胡道元. 计算机局域网(第三版). 北京:清华大学出版社,2002 (胡道元)

yibu chuansong moshi

异步传送模式(asynchronous transfer mode, ATM) 以分组交换为基础并结合电路交换的高速特性而形成的一种高速传送与交换技术。ATM本质上是一种高速分组传送模式。它将数据、图像和话音(言语)等信息分解成定长的信息块,并在信息块的前面装上含有地址等控制信息的信头以构成信元,并以信元为单位进行标记复用。由于只要获得信元头即可插入信息并传送出去,而且信息的插入位置是非周期性的,因而称之为异步传送模式。

异步传送模式(ATM)在信息格式和交换方式上与分组交换类似,而在网络构成和控制方式上与电路交换相似。

ATM克服了电路传送模式不能适应任意传输速度的缺点,又简化了分组传送模式中的协议,并由硬件对简化了的协议进行处理,且交换结点不对信息进行差错控制,从而大大提高了网络的通信处理能力。ATM靠标记(即逻辑通道号码)来识别通道,这种方式被称为标记复用或统计复用。

ATM采用长度固定的信元,可以采用硬件对信头进行识别和交换处理,以实现高速传输。来自不同信息源(不同业务和不同发源地)的信元汇集到一起,在一个缓冲器内排队,队列中的信元逐个输出到传输线路上,形成首尾相接的信元流。信元的信头中有信息的标记,说明该信元去往的地址,网络根据信头中的标记来传送信元。

ATM的信元结构

信元实际上就是分组,只是为了区别于X.25的分组(参见分组交换),才将ATM的信息单位叫做信元。ATM的信元具有固定的长度,为53B,其中5B是信头,48B是信息字段。信头的结构如图1所示。用户网络接口(UNI)和网间接口(NNI)上信头结构稍有不同。GFC是一般流量控制字段,用于控制同一接口上多个终端所发送的业务量,以减少可能出现的网络过载。VPI是虚通路标识符。虚通路是网络管理的单位,它由一个接口上的若干虚信

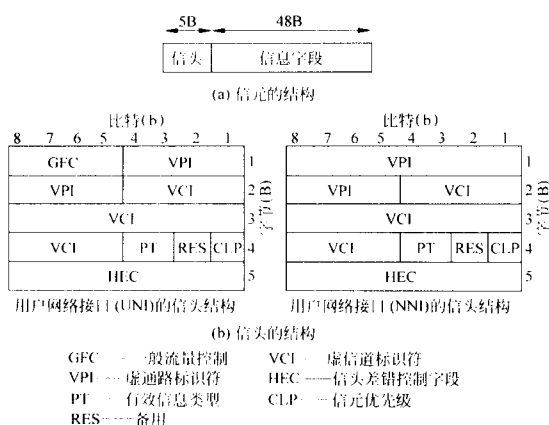


图1 异步传输模式(ATM)的信元与信头

道组成。VCI是虚信道标识符。虚信道是随呼叫的建立或释放而生成或取消的。虚通路(VP)与虚信道(VC)的关系以及它们与传输通道的关系如图2所示。在一个接口上用VPI和VCI这两个值就能识别一个呼叫。PT是有效信息负载类型字段,用来指示信息字段中的信息是用户信息还是网络信息。“00”表示用户信息。CLP用来指示信元丢失优先级。当网络拥塞时,可以丢弃CLP位为“1”的信元。当CLP位为“0”时,表示该信元有较高的优先级,为保证通信质量,网络不可丢弃这种信元。HEC是8b的信头差错控制字段,用以进行信头差错控制,防止因VPI、VCI出错而使信元串入其他用户终端造成干扰,其功能在物理层实现。RES为1b的备用字段,可用来增强信头的功能。

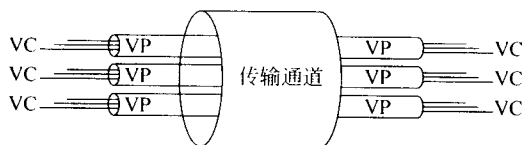


图2 虚信道(VC),虚通路(VP)和传输通道之间的关系

ATM的呼叫接续不是按信元逐个进行路由控制,它采用分组交换中虚电路(参见分组交换)的概念,即传送信息前先建立该呼叫的信元接续路由,该路由一直用到呼叫结束为止。

ATM的分层结构

ATM网的功能十分单纯,它只有与开放系统互连基准(参考)模型的第1层相对应的物理层和

ATM层,而与其他高层无关。信元在ATM层上传递,其标记在呼叫建立时分配,呼叫结束时释放,它是面向连接层的业务。

在ATM中,将开放系统互连基准(参考)模型的第1层细分为物理层、ATM层和ATM自适应层(AAL)。物理层承运信元流的负载。ATM层提供传送信元所需的最低功能。AAL层的主要作用是将第2层以上的用户信息分段装配成信元,并进行流控和差错控制。网络只提供到ATM层为止的功能。AAL层的功能由用户本身提供或由网与外部的接口提供。这意味着第2层以上能够原封不动地沿用已有协议的结构。这些层又可进一步细分为多个子层。其子层和它们的功能如图3所示。

高层功能		高层
会聚	CS	AAL
分段与重组	SAR	
一般流量控制 信头产生和提取 信元VPI/VCI翻译 信元复用和分用		ATM
信元速率解耦 HEC信头序列的产生和检验 信元定界 传输帧自适应 传输帧生成和恢复	TC	物理层
比特定时 物理媒体	PM	

图3 ATM协议模型

(1) 物理层进一步划分为两个子层:物理介质(PM)子层和传输会聚(TC)子层。PM子层的功能是在物理介质上正确地发送和接收数据比特流。物理层的功能与传输介质直接有关。TC子层将信元流变成能在物理介质上传送的比特流。

(2) 在ATM层中,主要实现信元的复用与交换,提供不同的业务质量并对其进行控制。

(3) ATM自适应层(AAL)介于ATM层与高层之间,它根据ATM层所提供的服务给用户以不同特性的通信业务的支持。根据高层是否建立连接,发送端到接收端是否需要比特定以及业务速率是否可变等,将AAL提供的业务定义为A、B、C、D和用户自定义5类。

AAL层分为两个子层:其上部为会聚子层

(CS),负责将业务数据变成 CS 数据单元。下部是分段与重组(SAR)子层,负责对 CS 数据单元分段或重新组装,并处理与业务分类有关的信息位。

以 ATM 技术支持的信元交换宽带网——ATM 网较为简单,除了第 1 层功能之外,交换结点不参与任何工作,因此它能提供很高的处理速率。从功能分布的情况来看,ATM 网和电路交换网有些相似,因此 ATM 可以说是融合了分组交换和电路交换的优点。

由于 ATM 克服了分组交换、帧中继交换的缺点,因此能够适应多种类型的业务,不论其传输速率高低、突发性大小、实时性要求和质量要求如何,都能向其提供满意的服务。因此国际电报电话咨询委员会(CCITT)在 1990 年就确定 ATM 为宽带 ISDN (参见综合业务数字网)的核心技术。ATM 是通信网络发展的方向,它可以把局域网和广域网统一起来,较好地解决数据、语音、图形、图像的综合传输问题。它可以在计算机之间支持一个统一的网络,而不管其物理位置如何。

参考文献

1. 程时端编著. 综合业务数字网. 北京: 人民邮电出版社, 1993
2. 李津生, 秋山稔. 综合业务数字网与异步转移模式 ISDN & ATM. 合肥: 中国科技大学出版社, 1993
3. Sidnie Feit. Wide Area High Speed Network. Macmillan Technical Publisher, 1999 (史美林)

yibu chuansong moshi juyuwang

异步传送模式局域网 (asynchronous transfer mode local area network, ATMLAN)

采用异步传送模式(ATM)作为数据传输协议构建的高速局域网。ATM LAN 被称为第三代局域网。对第三代局域网的典型要求包括支持多种特定类型服务,提供可增长的吞吐量及促进局域网和广域网(WAN)之间的互连。ATM LAN 能满足上述要求。它采用虚路径和虚通道实现多类型服务。通过增加 ATM 交换机结点和高速率设备可增大吞吐量。在广域网中使用基于信元的传送方式下,ATM LAN 能够无缝地集成 LAN 和 WAN (参见信元交换)。总之,ATM LAN 能提供多媒体应用所需的吞吐量、实时传输和质量保证。

ATM LAN 有以下几种应用类型: ①连至异步传送模式广域网(ATM WAN)的网关 将 ATM 交换机的作用类似一个路由器和集中器将 ATM LAN 连接到 ATM WAN; ②主干 ATM 交换机 用于连接其他 LAN 的一个 ATM 交换机或是由 ATM 交换机组成的局域网; ③工作站 ATM 将多媒体工作站直接连接到 ATM 交换机上。以上三种形式都是同一类型的配置。在实际应用中可将三者中的两者或全部结合,构成一个 ATM LAN。

图 1 是一个包含对 ATM WAN 连接的主干 ATM LAN 的例子。在这个例子中主干 ATM LAN 由 4 个 ATM 交换机组成。它们之间用 ATM 标准的传输速率(155/622 Mb/s)进行点对点连接。有三种其他类

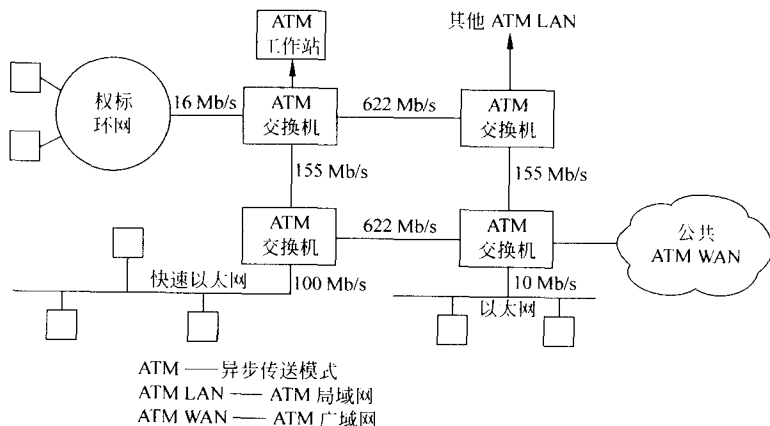


图 1 ATM LAN 的一个例子

型的 LAN 接在 ATM 交换机上。从 ATM 交换机到其他 LAN 的数据传送速率遵循该 LAN 的数据传输速率。例如,到以太网的速率为 10 Mb/s。因而 ATM 交换机必须有缓冲区和速率转换能力,同时还必须进行协议转换,以使 LAN 的介质访问控制 (MAC) 协议转换成 ATM 使用的信元流协议 (参见信元交换)。也有直接连接到 ATM 交换机的 ATM 工作站。

被连接在一起的不同类型 LAN 如何实现互操作是 ATM LAN 必须解决的重要问题。因为目前有千百万个以太网和权标环网在运行,用户不可能丢弃这些现有的局域网而立即升级到纯异步传送模式局域网。所以 ATM LAN 要取得成功的一个关键因素就是要有与传统局域网实现互操作的能力。为此 ATM 论坛定义了一种新的 ATM 业务,称为 ATM LAN 仿真。ATM LAN 仿真的目的就是使传统局域网站点能通过 ATM 网实现互操作,允许传统网络上的站点不需改变网络软件和应用软件就能通过 ATM 主干网传送数据,使传统 LAN 和 ATM LAN 实现共存。

ATM 论坛于 1995 年公布了 ATM LAN 仿真标准 1.0 版。1997 年又公布了 2.0 版。这些标准特别定义了 IEEE 802.3 以太网和 IEEE 802.5 权标环网 (参见局域网协议标准) 的 ATM LAN 仿真标准。规定了 ATM 网上的服务器如何与以太网和权标环网上的站点实现通信。图 2 是 ATM LAN 仿真的协议结构图。图中展示了连在 ATM 交换机的主机与连在传统局域网上的主机间的相互作用。需要特别指出的是连接到传统 LAN 上的主机没有任何的改变,包括介质访问控制 (MAC) 协议及逻辑链路控制 (LLC) 协议照常使用,并且在 LLC 上运行 TCP/IP 协议 (参见 TCP/IP 协议集) 及上层应用协议。为了通过 ATM 网络达到相互交换数据的功能,ATM-LAN 转换器需要有 LAN 仿真模块 (LANE), 由它完成介质访问控制 (MAC) 帧和 ATM 信元之间的转换 (参见异步传送模式)。LAN 仿真模块通过 AAL5 将 MAC 帧分成 ATM 信元,并和将收到的 ATM 信元重新组成 MAC 帧。ATM-LAN 转换器以通常方式连接到 ATM 交换机上,并成为 ATM LAN 的一部分。

ATM 主机也必须有一个仿真模块,才能与连到传统 LAN 的主机交换数据。ATM 主机上的仿真模块 (LANE) 从 ATM 适配层与 (AAL5) 接受 ATM 信元,将它转换成 MAC 帧并向上提交给 LLC 协议。因为 ATM 主机能够以远程低速 LAN 的格式接收和

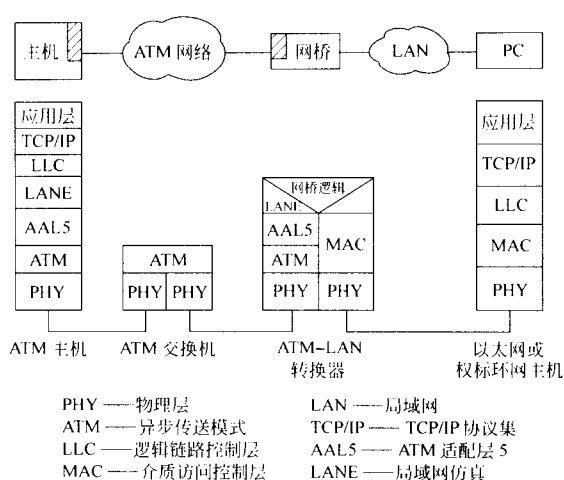


图 2 ATM LAN 仿真协议结构

发送 MAC 帧,它实际上仿真了那个低速 LAN。从低速 LAN 的一个端系统看,ATM 主机仅是有相同 MAC 地址格式的另一个端系统。

参考文献

高传善等译校. 局域网与城域网 (第 5 版). 北京: 电子工业出版社, 1998. (方起兴)

yichang chuli

异常处理 (exception handling) 在程序设计语言中用于描述异常与异常处置而用的语言机制。语言中提供的异常处理机制一般包括引发机制和处理机制两部分。

有几种解决异常处理的方案,一种是把异常处理看作针对非经常发生的事件 (不一定是错误) 的一种正常程序设计技巧,异常情况发生时,便由异常处理程序处理,处理结束时,控制还可以回到发生异常的位置。因此,也可以利用异常处理来实现一些修补工作。当修补结束时,又继续程序的正常执行。另一种则只限于异常情况发生了一些称作“错误” (或者是结束条件) 的事件。这便表示,在某一程序单位中发生异常情况时,执行就告终止,把控制转到异常处理程序,但以后就不再回到发生异常情况的位置,处理程序可以决定重新启动有关的程序段,但不是简单的恢复。无法恢复时,保留现场信息供分析用。

参考文献

徐家福. 系统程序设计语言. 北京: 科学出版社, 1983. (程虎)

yihou bianji

译后编辑 (post-editing) 为了提高可读性并修正错误,而对机器翻译系统输出的目标语言文本所做的编辑加工。这些工作包括:

(1) 多义词的选择 例如对“I am going to the bank”,系统输出“我去银行”或“我去岸边”;又如对“The girl wears red skirt often”,可能输出“这位姑娘常穿(戴)红裙子”。人可以根据上下文与汉语表达习惯进行校改。

(2) 译文句法错误的修正 例如对“I taught her how she should drive a car”,有的系统输出“我教她怎样她应该驾驶一辆汽车”。在汉语中副词“怎样”不能位于句首或分句之首,这句话很容易改成“我教她应该怎样驾驶汽车”。

(3) 语义错误的修正 例如对“I saw a man swimming on the bridge”,有的系统输出了“我看见了一个在桥上游泳的人”,通常,人不会在桥上游泳,这句话很容易改成“我在桥上看见了一个游泳的人。”

只要熟悉了机器翻译的译文风格和出错规律,即使不懂源语言或不参照源语言文本,也可以对目标语言的文本进行校改与编辑。不过,也有些问题必须同时对照源语言与目标语言的文本才能解决。

参考文献

冯志伟,杨平.自动翻译.上海:知识出版社,1987

(俞士汶)

yiqian bianji

译前编辑 (pre-editing) 在计算机自动翻译之前,由实用型机器翻译系统对源语言文本所做的一些编辑加工。

实际的源语言文件除了包含自然语言的语句之外,还包含图形、表格和各种排版符号,可以由预处理程序将这些信息及其位置分离出来,保存到另一个文件中。这个文件将为译文文件的自动排版和图表的自动插入提供參考。

译前编辑的大部分工作要由人完成,其目的是为了降低源语言分析的难度。这些工作包括:

(1) 标出专有名词 除少数例外,通用词典中一般不登录专有名词。尽管英语专有名词的第一个字母要大写,仍然无法区分句首的专有名词与普通名词(如 Jack 有两种可能:杰克与起重机)。汉语与日语的专有名词无形态标记,自动识别则更困难。

(2) 加上消解歧义的标志 以“The boy saw a girl with a telescope”为例,通常以句子为翻译单位的分析程序无法确定“with a telescope”是修饰 saw 还是修饰 girl。人可以根据上下文给出明确的标志。

(3) 补足省略了的成分 如需要对“I study physics and he chemistry”这句话的后一分句补上“studies”。

(4) 长句改短句 长句是机器翻译最难处理的,要求将总词数超过规定数目的长句改写成若干个传达同样信息的短句。

译前编辑工作规范的编制应尽可能使得只懂源语言的人就可以做这件工作。

译前编辑工作经验的积累与工作规范的完善将为受限语言的设计提供重要依据。受限语言是表达范围与表达能力受到限制的自然语言。这些限制涉及词汇、词义与句法结构,其目的是为了降低自然语言处理(包括机器翻译)的难度。为了减少人的记忆负担,可以开发某种受限语言的辅助写作系统。

参考文献

1. 冯志伟,杨平.自动翻译.上海:知识出版社,1989

2. 陈肇雄主编.机器翻译研究进展.北京:电子工业出版社,1992

(俞士汶)

yinzi fenjie

因子分解 (factoring) 两正整数 a, b , 若 b 能除尽 a , 则称 b 为 a 的因子, 记为 $b|a$ 。一个大于等于 2 的自然数若仅以 1 和自身为其因子, 称它为素数。任一大于等于 2 的自然数都能惟一地表成素数幂的乘积(算术基本定理)。设 $n \geq 2$ 为自然数, 则 n 可惟一地因子分解为 $n = p_1^{a_1} p_2^{a_2} \cdots p_s^{a_s}$, 其中 $p_1 < p_2 < \cdots < p_s$ 为素数, $a_i (1 \leq i \leq s)$ 为正整数, 称此式为 n 的标准因子分解式。由于计算机技术的发展和数论在密码学等领域的应用, 使研究大整数因子分解算法成为很活跃的研究课题(参见计算数论)。(裴定一)

yinhanshu qumian

隐函数曲面 (implicit function surface) 是由函数 $f(x, y, z)$ 的零等值面 $\{(x, y, z) | f(x, y, z) = 0\}$ 隐式定义的点集。与显式定义的参数曲面(如 B 样条曲面)不同, 隐函数曲面上的点一般不能直接计算。

常用的隐函数曲面有 Metaball、卷积和变分曲面、代数曲面等。与目前在几何造型中占主导地位的样条参数曲面相比较,隐函数曲面既可表达高阶连续的光滑外形,又能描述具有任意拓扑的几何形状。此外,隐函数曲面在曲线曲面求交、曲面光滑拼接、变形物体表示等方面也具有优势。隐函数曲面应用中的主要不便之处是其外形难以交互控制、曲面显示算法复杂。当前,有关隐函数曲面的主要研究内容包括参数曲面和代数曲面之间相互转化、分片代数曲面的理论和方法、形状交互与设计技术、曲面的多边形化与绘制等方面。自 20 世纪 90 年代以来,隐函数曲面已经在计算机动画领域中获得了较为成功的应用。随着理论和算法上的突破,隐函数曲面可望作为一种新形式的曲面描述方法,在 CAD 领域中发挥重要作用。

参考文献

Jules Bloomenthal, Chandrajit Bajaj, Jim Blinn, Marie-Paule Cani-Gascuel, Alyn Rockwood, Brian Wyvill, and Geoff Wyvill. Introduction to Implicit Surfaces. San Francisco, California: Morgan Kaufmann Publishers, Inc., 1997
(彭群生)

yingqian tuxiang chuli jishu

印前图像处理技术 (prepress image processing technology) 电子印前处理中使用的数字图像的色彩校正、层次校正、颜色模型的转换与显示、编辑与修改、特殊效果处理等技术。它是计算机数字图像处理技术在电子印前领域(参见电子印前处理)中的应用。在印前图像处理过程中,除了应用计算机数字图像处理技术之外,还应用了计算机图形学、计算机辅助设计、人机交互技术、色度学、印刷色彩学和彩色管理等技术。

印前图像处理技术在印前制版流程中,主要用于分色、版面设计、修版、拼版及创意等工艺。该技术的应用使人们摆脱了手工修版、手工拼版等繁杂的传统工艺,它是对传统制版工艺的革新。此外,还使许多传统制版工艺不能实现的特殊效果得以实现,使彩色制版工艺的质量和生产效率得到提高。印前图像处理技术是在 20 世纪 80 年代中期,随着与电子分色机一起使用的整页拼版系统的出现而发展起来的一门新技术,并随着彩色桌面系统的发展而得到普遍使用。

印前图像处理技术的主要内容包括:色彩复制、修版、拼版、挂网和分色处理等。

色彩复制 又称为颜色复制(参见颜色复制),指的是在电子印前处理中,将扫描仪、数码相机等设备获取的和(或)计算机处理得到的数字彩色图像,通过数字或模拟的手段,在打印机、彩色印刷机等硬复制设备上真实地再现的技术。

修版 在计算机屏幕上通过直接修改图像像素来改变此图像的局部或总体状态的技术。计算机修版技术起源于对暗房修版工艺的模仿,因此通常都提供相应的数字修版工具来模仿如刷子、尖笔等的功能。此外,还引入了图像分层的概念,使图像修版处理的可编辑性得到提高。修版技术曾经是拼版系统的主要实现技术,它主要用来清除经过扫描得到的数字图像上的脏点和划痕等。借助于数字图像处理的理论和方法,还可以通过修版来实现图像的某些特技变换效果和清晰度控制。

拼版 将不同的版面元素(图像、图形、文字)结合起来,形成一张符合出版要求的版面。其内容主要包括:版面元素的定位、缩放、旋转;彩色图像的颜色校正、裁剪拼接;装饰图案、文字的设计与生成等。在电子技术引入印刷出版行业之前,拼版是通过照相和暗房技术用手工完成的。现在,彩色电子拼版已经可以在个人计算机上完成,不但成本有所下降,版面质量及生产效率得到提高,而且为拼版技术增加了新的内容,使其从单纯的彩色图像处理发展为包含有图形图案设计、彩色管理、图文特技和艺术图像生成在内的综合性的计算机辅助出版技术。

挂网 在二值化设备或非连续色调的设备上输出具有丰富灰度等级或彩色的图像的技术。传统的方法是用一块小面积(如 $0.02\text{mm} \times 0.02\text{mm}$)上不同的像素图案来模拟该小面积的总亮度值,这种方法叫**半色调加网**。采用半色调加网产生灰度图像的效果是由于人眼观察时对局部空间的灰度进行了平均的缘故。挂网的优劣直接影响彩色输出的质量。因此,一些研制生产彩色印刷设备的大公司都花了很大精力来研究改进挂网技术,以得到高质量的彩色输出。目前,比较流行的挂网技术有 3 种:有序抖动技术,调幅网技术和调频网技术,特别是 20 世纪 90 年代出现的调频网技术,它利用网点的疏密程度来表示不同的灰度级,而网点大小固定不变,其最大优点是克服了传统挂网过程中因重叠 4 张分色片而产生的龟纹现象或花瓣图案,受到了印刷界的欢迎。

分色 在彩色印刷的电子印前处理中,彩色原稿经扫描仪输入后是采用 RGB 颜色模型表示的,而硬复制输出设备一般都使用 CMYK 颜色模型。分

色就是把彩色图像分解成青、洋红、黄、黑 4 种颜色的灰度图。它首先把彩色图像中的各种颜色由 RGB 颜色模型转换成 CMYK 颜色模型,然后再将彩色图像储存为青、洋红、黄、黑 4 种颜色的灰度图。如果在原有 CMYK 颜色的基础上再加入淡蓝绿色、淡红紫色和其他颜色,即使用 6 色或 7 色的彩色喷墨打印机输出彩色图像,则分色处理的过程会更复杂一些。

参考文献

1. 风子. 电子分色制版工程. 北京: 印刷工业出版社, 1992
2. Kieran M. Desktop Publishing in Color. BANTAX BOOK, 1991 (阳振坤 周秉锋 张福炎)

yinshuati hanzi shibie

印刷体汉字识别 (printed Hanzi recognition) 通过扫描仪将印刷文本输入计算机,并将其中的汉字转换成国标代码的过程和技术。

印刷体汉字识别要解决各种各样实际应用中出现的上万个不同印刷体汉字的识别问题。需要解决的难题有:

(1) 巨大的识别汉字集合, 简体 6 767 个, 简繁混合 27 484 个。

(2) 不同的字体的多样变化, 基本字体有宋、仿宋、楷、黑、圆、隶书、魏碑等主要字体, 以及各种变化和美术字体, 还有各种字体所有的不同字形, 总数达百种以上。

(3) 不同大小字号的汉字, 1~7 号, 还经常出现在同一页面。

(4) 汉字文本中除包括必要的数字和标点符号(数十个)外, 现代文档均为汉、英混排文本。因此, 汉字识别必须同时解决汉、英双语识别和汉、英混排文档识别的特殊文字切分、语种判别问题。

(5) 实际印刷文本经常出现的各种噪声干扰, 汉字的模糊粘连或笔画断裂不全, 汉字字形的拉伸切变等几何形变的低质量汉字的识别。

(6) 摄像设备的便宜和普及, 摄像输入文档的文字识别需求大量增加。

除了上述单字识别的问题以外, 所有印刷汉字识别系统还都必须解决好字符串的单个字符切分问题。由于字符粘连和断裂, 以及多种字符(数字、标点、英文)的混排问题, 使得字符切分与字符识别达到几乎同等困难和重要的程度。

文字识别技术的研究和发展, 最重要的表现为

识别的鉴别能力的不断提升及对上述变化的适应能力大为加强, 即识别的鲁棒性不断提高。为此, 不仅进行特征的选择和优化, 而且在分类器上也利用二阶非线性分类器, 如 MQDF 等。实验证明, 这些统计识别方法较好地解决了印刷汉字识别的问题(参见汉字识别基本方法)。

印刷体汉字识别研究经历了从单字体汉字识别开始, 经多字体汉字识别, 识别字体不断增加, 识别汉字的字符集合不断扩大, 汉字识别率, 尤其是对印刷质量低下的汉字的识别率得以不断提高的过程。近来, 输入设备从扫描仪向摄像机发展, 识别图像从二值图像到灰度、彩色图像的识别。目前开发出的识别系统已经不仅能够同时识别几乎所有字体, 而且对实际低质量的印刷文本和摄像印刷文字也有很高的识别率。识别率可以达到 99% 以上, 基本满足实际文本自动输入的需要。

我国有五千年悠久的文明历史, 大量的古籍文明也需要也可能登上现代信息化的列车。因此, 对古代汉语近 5 万个古籍汉字文本进行识别输入计算机的任务也摆到了我国科学工作者的面前。

(丁晓青)

yinshua wenben banmian fenxi

印刷文本版面分析 (printed page layout analysis) 对印刷文本版面的排版格式进行自动分析、切分和标识的过程和技术。

由于书面文字信息均按照一定的版面排版印刷成为文本图像, 它包含了印刷文本记载的字符串有机组成的信息的表述。因此, 书本、杂志、报纸等数字化时得到的是整个页面的图像。对于印刷文本的识别, 首先需要的是从整个页面图像中将文字块分割出来。将文字块从复杂的版面中分割出来的过程, 需要进行版面分析。即经过版面分析, 将文档页面的各种图文属性内容进行分区分割, 再将文字块从版面中分割出来。然后, 对文字块图像进行文字行切分和文字切分的逐步处理, 最后获得单个字符的图像, 再针对单个字符图像进行字符识别, 获得单字的识别结果。将单字识别结果汇总, 得出整个篇章和文档的数字化结果。因此, 除了要求正确的单个字符的识别外, 印刷文本的版面分析和文本图像中字符图像的正确切分是文档(文本)正确识别的前提。

印刷文本是具有一定版面排版格式的印刷品。印刷文本的自动版面分析是对印刷文本版面的排版

自动进行区域分割、属性判决和标识的过程。实际的印刷文本是由若干不同属性的区域,如文本、图像、图形、表格等组成。文本区域则包括标题、作者、正文、公式、注释、页码等。正文块又可分为单栏、双栏、三栏等编排格式。文本块之间用空白条、直线、装饰线、花边等隔开。这些版面不同区域的分割和属性的标注,以及复杂版面中文本块间的阅读顺序和篇章分割等,都是版面自动分析和理解的重要内容,它是文本识别的重要前处理,更是文档全信息数字化和文档自动索引和标注生成的重要手段。

复杂版面(如报纸版面见图1)的自动版面分析和理解,由于版面的复杂和多变性,是一个十分困难的问题。一般有按自上而下(主要利用投影的方法)和自下而上(逐级合并的方法)两种算法。近来又按照利用形状分析和纹理分析将算法分为两类。有多种新算法提出,如基于形状的方法有多层次置信度指导下的自底向上的版面分析算法,基于纹理的算法有统一的隐马尔可夫模型(HMM)版面分析框架及算法等。



图1 复杂报纸的版面分析

对于文档信息而言,除了字符表示的文字信息以外,版面排列、标题、栏目以及字符大小和字体等,也往往包含有重要的信息。保留整个版面信息的文档数字化称为全信息数字化,是文档数字化的重要内容。

参考文献

1. 陈明,丁晓青,吴佑寿. 多层次可信度指导下的自底向上的版面分析算法. 模式识别与人工智能, 2003, 16(1): 1~6
2. Chen Ming, Ding Xiao qing, Wu Youzhou, Unified HMM-based Layout Analysis Framework and Algorithm. 科学通报, 33(6) (丁晓青)

yinzhiban ceshi

印制板测试 (printed circuit board testing)

以多种专用设备对裸印制板和已装元器件印制板的印制线通断的正确性、绝缘性能,以及逻辑功能等进行的全面测试。万用表、示波器是传统的印制板的测试工具和仪器。随着电子工业的发展,印制板的复杂程度越来越高。基于各种测试方法的印制板测试仪已日益增多,从功能来分类主要有裸印制板测试仪、已装印制板测试仪和电路板维修测试仪三大类。其中已装印制板测试仪又可分为在线测试仪(包括故障分析仪)、功能测试仪和综合测试仪。

裸印制板测试 裸印制板测试主要是判断印制板印制线通断的正确性及其绝缘性能。其测试方法通常采用一个进行快速扫描的电桥,当被测两点之间的电阻小于量程标值时电压比较器输出1电平,大于量程标值时输出0电平,这样就可判断被测两点之间的通断情况。一般测试用电压为0.1 V~10 V,绝缘测试时需250 V~500 V(甚至需1 500 V)。裸印制板测试仪一般由微型计算机、测试电路和测试针床组成。测试针床是这类测试仪的关键部件,弹簧触针又是测试针床的关键零件。触针上加有0.98 N(0.1 kgf)的测量力,触针应有50万次以上的寿命。触头的形状可根据被测板的类型选择三棱柱形、皇冠形等形状。早期的这种测试仪一般采用专用针床,后来多采用通用针床。通用针床是按标准网格在每一个网格点上放置一个弹簧触针,在更换被测印制电路板品种时采用这种网格格式触针阵列可减少针床的制作费用。裸印制板测试仪向着网格格式通用针床、高测试点容量、高测试速度、高压绝缘测试的方向发展。

印制板在线测试 印制板在线测试就是测量印制电路板内印制导线、跨接线连线的正确性和每个元器件的参数值及功能的正确性。根据印制板内元器件的类型,在线测试又分为模拟在线测试和数字在线测试。模拟在线测试是在被测电路板不加电的情况下,借助于运算放大器对电路板内的模拟元件

进行隔离和测试,这样可以测量其装配的正确性和元件的参数值。数字在线测试是被测电路板加电的情况下,用反驱动技术将测试图形强制加到被测数字集成电路的输入端,根据输出端的响应,判断集成电路逻辑功能的正确性。

在线测试仪也采用针床测试夹具,以便使用在线隔离技术准确测量单个元器件。在线测试可精确定位有故障的器件,且测试程序的编制比功能测试简单得多。

故障分析仪是一种低成本的在线测试仪,这类仪器不具备对数字电路的在线测试能力。它采用节点阻抗的方法来测量每两个节点间的阻抗值,并将测量值与从一块好的印制板上读出的数值相比较。对于一些有并联支路的元件可采用在线保护技术把被测元件与电路板上其他元件孤立起来测量其参数值。故障分析仪可以检测出印制板上的短路、开路,元器件的错装、漏装、方向颠倒,元器件数值超差等制造故障。该类仪器的价格约为大型在线测试仪的 $1/3 \sim 1/7$,而故障覆盖率仍可达 $85\% \sim 90\%$ 。

印制板功能测试 将印制板看作一个完整的模块,测试其功能。测试时给被测电路板加供电电源,使其处于正常的工作状态,从电路板的输入端输入激励信号,检测输出响应。为了故障定位,要输入适当的测试码,这样才能将内部故障传递到输出端。

功能测试的编程比较复杂,必须详细了解被测试电路板上元器件的电气特性,采用大内存容量的高速计算机。通常这类测试放在电路板装配流水线的最后,用来检测产品的整体功能,故障覆盖率可达 95% 左右。

后来出现的一种**印制板综合测试仪**同时具有静态在线测试和动态功能测试两种测试功能。

印制板维修测试 指不是批量生产时的印制板测试,测试的目的是找出有故障的器件,修复印制板。印制板维修测试仪一般不采用针床结构,而采用“夹子”或探笔的方式,结构紧凑,便于携带。

数字电路印制板的维修测试可以使用通用的测试仪(如逻辑分析仪、仿真器、微型计算机故障诊断仪等)。专用的印制板维修测试仪有两大类,一类是具备在线测试能力的维修测试仪,它通常有 40 至 64 个数字测试通道,每个通道至少有 $1\text{ kb} \times 4$ 的通道存储器。可测试小规模、中规模、大规模集成电路以及半导体存储器等器件。这类测试仪有的还带有模拟测试通道,可测量电压、周期、延时等参数。另一类是采用 $U-I$ 曲线法的维修测试仪。这种测试仪

使用时给被测元件加上交流电压信号,用被测元件两端产生的电压降控制示波器的水平偏转,用通过被测元件的电流控制示波器的垂直偏转,这时示波器荧光屏上出现的就是元件的电压-电流关系曲线,通常称为 $U-I$ 曲线。由于每种元件都有各自的 $U-I$ 曲线,因此 $U-I$ 曲线也称为特征波形。通过对特征波形的分析、比较就可方便地判断元器件的好坏,找到电路板故障所在。该类仪器的特点是测试时被测电路板是不加电的,各种类型的器件都可用特征波形法判断其好坏。

基于新的测试方法的仪器,比如利用数据压缩技术的特征分析仪,采用红外线技术的电路板缺陷测试仪等,都可对印制板进行维修测试。(桑光道)

yinzhiban sheji

印制板设计(printed circuit board design) 根据计算机部件的逻辑框图或电路连接图用印制电路技术实现有关元器件之间的互连,并达到预期功能和性能的设计方法和过程。

在计算机工程设计中,插件板与底板是印制板(PCB)设计的主要任务。插件板一般是具有独立功能的部件(如处理器、存储器、输入输出接口、通信部件等),或者是具有一定逻辑关系的组合部件。底板则是实现插件间电气连接与电源馈给以形成系统或大部件的重要结构。在较复杂的计算机系统中,往往还要进行一些辅助板或附加板(如匹配板、转接板、扩展板等)的设计。

印制板设计的主要内容

结构设计 结构设计既要解决PCB与其他结构件相互关联的外部相关结构问题,又要确定PCB的内部结构问题。

外部相关结构设计的主要内容有:①外形尺寸(长、宽、厚度)及精度;②在整机中安装的结构形式,有关的安装尺寸及精度;③与其他结构件(如插件围框、连接器、插拔器、特殊器件的紧固件、散热片等)的配合方式及尺寸、精度;④为PC板或整机加固而设计的安装孔的尺寸、位置和精度;⑤满足某些特殊要求的屏蔽结构。

在进行外部相关结构设计时要考虑PCB结构与整机结构的协调,如插件与机箱、框架的配合,插件与底板的配合,特别是插件与连接器(插头、插座)的配合,插件与辅助功能结构(如插拔机构)的配合等。此外,还要考虑整机综合性能的要求,可能涉及的设计项目有机械加固设计、整机热设计、电磁

兼容性设计、可靠性设计、可维修性设计以及标准化设计等。

内部结构设计的内容为:确定 PCB 的层数(单面板、双面板、多层板),根据多层板的总厚度确定层间绝缘介质的厚度分配,确定板内各种焊盘、过渡盘、焊接孔、过线孔的尺寸及精度要求等。进行内部结构设计时首先要综合电气性能需求、布线设计要求等各方面因素来确定 PCB 的层数,应力求减少层数以降低成本和提高可靠性。

对较复杂的电路,特别是器件工作频率较高的电路进行 PCB 设计时,电源、地线网络的安排对改善馈电特性,调节信号线的传输阻抗,隔离线间干扰,提高布线的布通率等方面都有很大影响。

在设计印制底板时,为了整机运行的安全、可靠,通常避免在表面层布线。

内部结构所涉及的焊盘、孔等尺寸,可根据有关的标准、规范,结合 PCB 加工的工艺水平来确定。

布局设计 利用交互或自动的方式合理摆放欲在 PCB 板上安装的元器件。主要过程为:首先在 PCB 板的表面层(顶层或底层)划分布局区域与禁止布局区域,然后用手动方式摆放有特殊要求的元器件,用自动方式摆放其余的元器件,最后根据电特性要求、可测试性要求、布线难易情况等对布局结果进行交互式优化调整。有特殊要求的元器件一般是指安装位置固定的元器件、安装在 PCB 板四周边缘位置上的接插件、需要多种工作电源的元器件和引脚数超过 100 的高集成度芯片等。

布局设计的基本原则如下:

①满足基本的功能要求。超过 100 MHz 的高速总线通常对某些关键信号有特殊的传输延时要求,高速信号处理芯片也常会要求与外围芯片的某些关键连线保持相等的传输延时,所以,相关元器件的摆放必须考虑这些限制,以免布线满足不了要求。②数字信号元器件与模拟信号元器件分区放置,以降低干扰。③有些元器件同时采用 5 V、3 V、1.6 V 的工作电源,应尽量把工作电源相同的元器件靠近摆放,以节省布线空间。④为了有利于元器件之间的互连,应调整元器件间的相邻位置,以提高布通率;元器件的排列方式要有利于通风散热,温升高的元器件放在易于通风散热的部位;元器件的安装焊盘要落在规定的网格坐标点上。⑤易于受干扰的元器件应远离大功率信号器件及大电流信号线,输入输出信号多的元器件应尽量靠近对外互连的接插件,需要隔离、屏蔽的元器件要根据实际需要进行特殊的安

排(例如相对集中)。⑥便于测试。⑦便于维修。

布线设计 利用交互或自动的方式把 PCB 板上应互连的元器件用印制导线连接起来,是印制板设计的核心内容。

布线设计的主要过程如下:在 PCB 板的各层划分布线区域与禁止布线区域;确定设计规则(定义印制导线的宽度、金属焊盘与非金属焊盘的形状与尺寸、过渡孔焊盘的形状与尺寸),确定物理规则或称间距规则(导线与焊盘的间距、焊盘与焊盘的间距、导线之间的间距、导线与金属图形的间距),确定电气规则(定义 ECL 信号、定义差分对、定义信号延时等);首先完成电源信号和地信号的连接,然后用交互或自动的方式完成其余的连线;根据工艺特点对连线进行自动优化处理,比如线平滑、线均匀、45°拐角处理、删除多余过渡孔等;设计丝网印图形;进行设计规则检查,排除错误;完成生产加工文件后处理工作,即分层定义要加工的金属图形、焊盘阻焊图形、元器件的丝网印图形、印制板标记的丝网印图形和钻孔图形,定义 D 码表,最后输出这些图形的工艺文件并提交给生产厂家。

布线设计的基本原则如下:

①要保证各个互连线正确、可靠地连通,必要时可采用冗余多点连接。②对于有大电流通过的印制导线,必须考虑要有足够的宽度,以保证其承载电流负荷的能力。③对于有大电位差的相邻印制导线,必须要注意保持适当的间距,防止击穿。④对于数字集成电路,在 PCB 布线时要特别注意减少噪声干扰,基本措施是:利用接地平面(包括电源网络和地线网络)降低印制导线的特性阻抗,避免出现印制导线平行耦合情况或尽力减小印制导线的平行耦合长度,尽可能减少互连引线的长度,对一些关键信号可以附加地线屏蔽。⑤对于线性电路,要特别注意控制反馈,避免产生振荡。⑥对于静电敏感性器件,在设计时要考虑适当的防静电损伤措施。⑦PCB 布线设计要与工程化设计紧密配合,妥善地处理好有关的工程化问题,如电源滤波(电源与地线之间采用分布式的、覆盖足够宽的工作频率的滤波电容),逻辑电路空闲输入端的处理(为防止干扰,将空闲输入端接地或高电平),阻抗匹配,防静电保护等。⑧对一些电容负载能力差的电路,在布线时要注意控制印制导线的分布电容(减少导线的宽度和长度,尽量布在与接地平面距离远的层次上)。

印制板设计原则 印制板设计质量的高低,对计算机的性能(特别是可靠性)有重大的影响,甚至

影响到计算机的正常功能能否得以实现。要高质量地完成 PCB 设计,必须将计算机整机工程特征、各种元器件的性能特征、PCB 的制造工艺特征、工作环境特征等多方面的相关因素加以有机的综合,在设计过程中统筹考虑,有针对性地处理好有关的工程技术问题,使 PCB 组装的部件和整机的功能、性能得到有效的保障。

印制板设计方法 早期的印制板设计是采用手工绘图的方法进行的。后来,随着计算机图形学和计算机辅助设计技术的发展,电子产品自动设计系统(EDA)开始应用,取代了手工设计方法。EDA 可从部件描述开始,按照一定的规则导出逻辑图。设计人员以 EDA 的输出文件、结构设计文件和有关标准规范作为设计输入条件,利用先进的软件工具,通过人机交互方式完成印制板的外形结构设计、元器件布局、印制板布线、分析与验证、设计规则检查和工艺后处理,最后输出生产加工文件。

20 世纪 90 年代以来,芯片制造技术、表面贴装技术越来越成熟,印制板的密度和板上信号的速度有了极大提高,高速信号电路板的设计成为 PCB 设计的焦点。高速信号电路板通常要解决串扰问题、信号完整性问题、散热问题和电磁兼容问题,未经过分析与验证的高速信号电路板很难保证一次设计的正确性,每一类问题都要依靠专门的软件工具进行分析验证,针对分析出的问题去修改相应的布线或布局。分析与修改的过程可能重复很多次才能最终得到满意的设计结果。(宫世友 万芙蓉)

yinzhiban zaixian ceshi

印制板在线测试 (in-circuit testing of printed circuit board) 借助在线测试夹具,对印制板(PCB)上的电子元器件逐个进行功能测试。它与印制板功能测试的区别在于它不将印制板看作是一个逻辑整体来进行测试;它与单个元器件测试的区别在于它仅测试元器件的功能或量值,而不测试其他交直流参数和特性。

印制板在线测试起源于 20 世纪 70 年代末、80 年代初。当时,随着大规模集成电路 (LSI)、超大规模集成电路 (VLSI) 的出现,印制板越来越复杂,传统的功能测试所采用的测试图形集和故障字典已难以迅速准确地检测并定位故障,不能满足印制板大批量生产的测试需求,印制板在线测试也就应运而生了。

印制板在线测试根据板内元器件的类型可分为

印制板模拟在线测试和印制板数字在线测试。前者是在被测板不加电的情况下,借助运算放大器对板内模拟元件逐个进行隔离和测试,它可测量该元件生产装配的正确性和量值。后者是在被测板加电的情况下,用具有强迫驱动能力的激励信号源,将测试图形强制加到被测集成电路 (IC) 的输入端,然后根据回收到的输出端响应,判别该数字 IC 逻辑功能的正确性。

印制板在线测试具有强驱动、电隔离及时间保护的特点。印制板在线测试的本质是对印制板上各单个元器件的测试,为了排除板上与被测元器件互连的其他元器件造成的影响,在线测试必须采用特殊的隔离与驱动方法。

数字在线测试中,由于电路互连,板上某被测 IC 的输入端往往同时是前级 IC 的输出端,因此,被测 IC 输入端的初始逻辑状态是由前级 IC 输出端的逻辑状态确定的。要在被测 IC 输入端施加测试所需的输入激励信号,该信号必须具有足够的强迫驱动能力,不管前级处于何种逻辑状态,都将它强迫驱动到测试所需的激励状态。这就是**强驱动**(也称为**过驱动**或**反驱动**),强驱动源则称为驱动器。

强驱动同时实现了被测电路与前级电路的电隔离。这种以强驱动实现隔离的功能还用于测试总线器件时隔离其他总线器件的输出,测试集电极开路门 (OC) 时隔离其他 OC 门的输出,此外还用于切断逻辑闭环等场合。

强驱动的实现需要足够的双向驱动电流。以图 1(a)所示的双极型 TTL 电路为例,要使前级电路输出端原来的低电平逻辑状态,强驱动到逻辑高电平,意味着驱动源要向处于深饱和和工作状态的 TTL 输出晶体管 T_3 注入几十甚至几百毫安的驱动电流 I_1 ,使其脱离饱和区而进入线性放大区(如图 1(b) T_3 管特性曲线上工作点由 A 到 B)。反之,要将前级电路输出端原来的高电平逻辑状态强驱动到逻辑低电平,则驱动源必须能吸收流经 R_c 的足够大的驱动电流 I_2 ,使它在 R_c 上形成的电压降足以保证输出端从逻辑高电平降至逻辑低电平。可见,在强驱动时前级电路要注入或拉出很大的强驱动电流,最大可达几百毫安,以迫使电路处于一种非正常工作状态。基于电路安全的考虑,只允许在尽量短的时间内维持这种工作状态。该时间长短与强驱动电流的大小有一定的关系,如图 2 所示。电流越大,允许时间越短。对于不同的电路类型及不同的强驱动脉冲占空比,安全曲线是不同的。一般在驱动电流

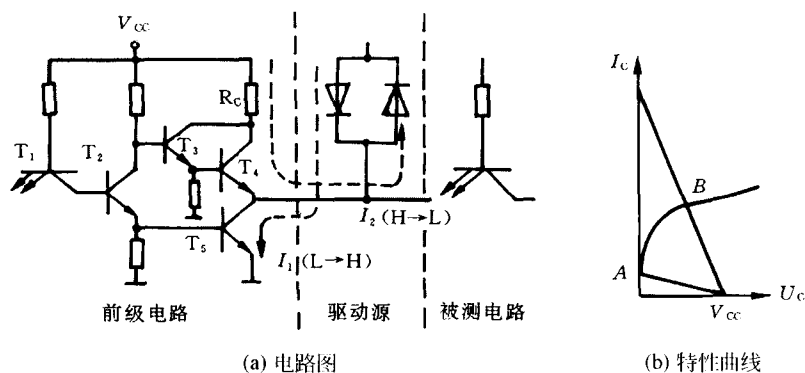


图1 TTL 电路强驱动原理

小于 100 mA 或驱动时间小于 50 ms 情况下器件是安全的, 否则可能引起电路慢性的或永久性的损坏。为此, 在线测试设备都对强驱动的时间保护有严格的规定和严密的控制。

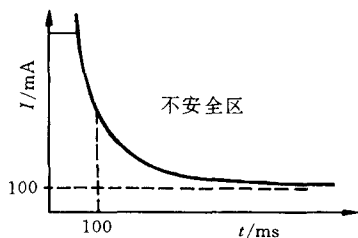


图2 强驱动电流安全曲线

模拟在线测试中, 隔离功能是靠运算放大器的虚地特性和隔离引脚实现的。以三线法电阻测试 (见图 3) 为例, 由于放大器输入 A 点为虚地, G 点采用隔离引脚到地, 使 R_2 两端处于同电位, 无电流通过, 从而隔离了 R_1, R_2 并联通路对 R_x 测量的影响。

为了实现对印制板内各单个元器件的测试, 在线测试必须借助于专用的测试夹具以实现与板内各元器件的电气接触。

在大中型在线测试设备中, 适用于整板一次性自动测试的测试夹具通常采用针床夹具, 如图 4 所示。测试设备通过探针向被测元器件的输入端施加激励信号, 并从输出端回收输出响应, 通过程序自动对板上元器件逐个进行测试。不同的印制板必须制作不同的针床夹具。测试中, 针床夹具通常采用真空系统来实现与被测板的吸合接触。采用针床夹具的印制板在线测试还具有电路板连线

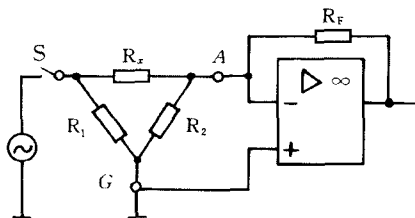


图3 模拟在线测试的电隔离

的通断测试功能。

小型在线测试仪上采用的在线测试夹具称为测试夹, 它通过电缆与测试仪相连, 由人工逐个夹住被测 IC 进行测试。

印制板在线测试的主要优点是: ①由于本质上是单个元器件的测试, 故测试编程简单, 测试容易; ②故障查找迅速, 定位准确; ③测试速度快, 产量大。缺点是: ①不能检测印制板的时序故障, 故障覆盖率较功能测试低, 为 90% 左右; ②针床测试夹具价格昂贵, 不适用于多品种小批量测试; ③由于强驱动的限制, 测试频率低于功能测试。

印制板在线测试已逐步形成一套完整而成熟的软

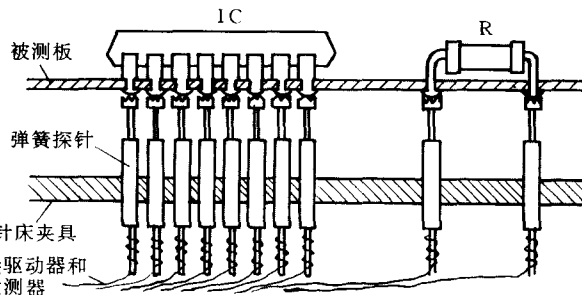


图4 针床夹具示意图

硬件测试技术,如各种测试电子线路、测试控制方法、硬件描述语言(HDL)以及各种自动测试程序生成方法(ATG)等,目前它已成为印制板主要的测试方法之一,广泛应用于印制板的大批量生产性测试中。

近年来随着 IC 封装和 PCB 组装技术的发展, VLSI 的引脚间距已小到(如小到 0.203 mm)难以再用在线测试探针方式实施接触。表面安装技术(SMT)及其他更先进的焊装技术,如多芯片模块(MCM)、板上芯片(COB)与更多层的超薄化 PCB 等技术的采用,又限制了在线测试的有效应用。1985 年以来,出现了一种新的测试技术——边界扫描测试技术,它通过边界扫描寄存器解决了采用 SMT, MCM, COB 等技术的器件的实体存取问题,其应用日益广泛。由于传统器件和采用边界扫描技术设计的器件将在相当一段时间内存并,因此,印制板在线测试技术和边界扫描技术的互相结合和支持仍将是近期内印制板生产性测试的主要方法。

参考文献

程旺,陈开华. 数字电路板在线测试. 计算机工程与应用,1986,7: 7~14 (陈开华)

yinzhiban zhizao

印制板制造 (printed circuit board manufacturing) 印制板是一种用以安装电阻、电容、电感、集成电路等元器件并提供它们之间的相互电气连接的部件。

国家标准 GB 2036 中,对印制、印制电路、印制线路及印制板等的定义描述如下。

印制: 采用某种方法,在一个表面上再现图形的工艺。

印制电路: 在绝缘基材上,按预定设计形成的印制元件或印制线路以及两者结合的导电图形。

印制线路: 在绝缘基材上形成的导电图形,用于元器件之间的连接,但不包括印制元件。

印制板: 印制电路或印制线路成品板的通称。它包括刚性、挠性和刚挠结合的单面、双面和多层印制板。

仅一面上有导电图形的印制板为单面印制板,两面上均有导电图形的印制板是双面印制板。由多于两层导电图形与绝缘材料交替黏结在一起,并要求导电图形互连的印制板称多层印制板。它包括挠性和刚性多层印制板以及刚性和挠性结合的多层印制板(刚性、挠性是指印制板的绝缘基

材而言)。

印制板制造的工艺流程,根据印制板的分类而有所不同。

单面印制板制造工艺流程 此流程参见图 1,其中光化学图像转移工艺成本较高,通常用于生产多品种、小数量的单面板。

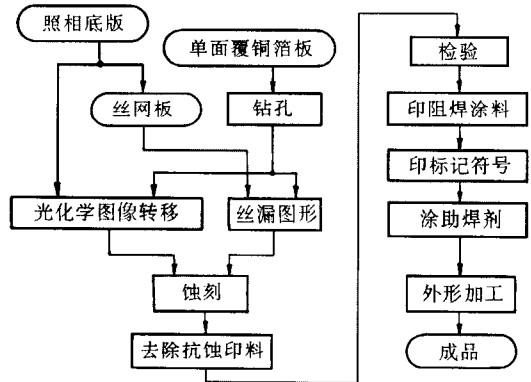


图 1 单面板工艺流程图

双面印制板制造工艺流程 现代印制板制造中,多数已采用计算机辅助设计(CAD)和计算机辅助生产(CAM)。由 CAD 可以产生照相底片(光绘)、数控钻孔数据和自动检测数据等。由 CAM 可以监控印制板制造过程中的工艺参数,以保证产品质量。图 2 是双面印制板工艺流程图,其中图像转移可以采用丝网耐镀印料、光敏干膜、液体光敏抗蚀剂等工

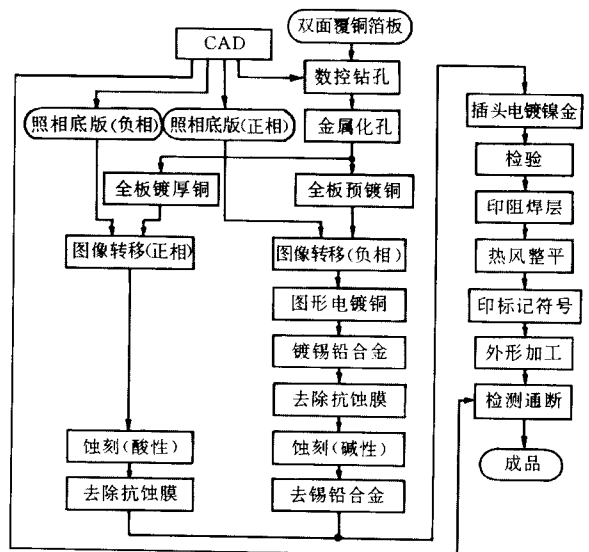


图 2 双面印制板工艺流程图

艺。现代印制板制造中,图像转移已逐渐由光敏干膜转向液体光敏抗蚀剂(湿膜)过渡。

多层印制板(刚性)制造工艺流程 多层印制板与双面印制板的制造工艺有共同之处,其特殊性主要体现在:①必须有定位系统,如照相底版定位、图形转移定位、层压定位、钻孔定位等;②需要预制内层电路薄片;③需要预浸渍环氧树脂玻璃布片,用它将两块内层电路薄片黏结在一起并实现电绝缘;④增加层压、凹蚀等生产工序;⑤各生产工序的质量控制更加严格,且成品必须进行电测通断;⑥单、双面印制板的底版,可以通过布线贴图或墨绘底图后照相产生,也可以由 CAD 产生;但多层印制板尤其是高密度、高层数时的底版,为保证定位精确,必须由 CAD 产生。

上述三种印制板工艺流程,均基于铜箔腐蚀原理。至于挠性、刚挠结合、高频及耐高温材料制成的各类印制板,尚有其不同的制造工艺。

参考文献

《电子工业生产技术手册》编委会编. 电子工业生产技术手册(12)——通用工艺卷. 北京: 国防工业出版社, 1992 (方简秉)

yingshe

映射(mapping) 两个集合元素之间的一种对应规则。映射有时又称函数。

设 X 和 Y 都是集合。若对每个 $x \in X$, 根据对应规则 f , 都有惟一的一个 $y \in Y$ 相对应, 则称 f 为一个从 X 到 Y 的映射或函数, 记为 $f: X \rightarrow Y$ 或 $X \xrightarrow{f} Y$, 并把 y 表示为 $f(x)$, 即 $y = f(x)$, 这时称 y 为 x 在 f 下的象, x 为 y 关于 f 的原象, 并称 X 为 f 的定义域, Y 为 f 的值域空间。

例如, 取 $X = \{a, b, c\}$ 及 $Y = \{0, 1, 2\}$ 。若对应规则 f 定义如下

$$a \mapsto 0, \quad b \mapsto 0, \quad c \mapsto 2$$

则可得到一个映射 $f: X \rightarrow Y$ 使 $f(a) = f(b) = 0$ 且 $f(c) = 2$ 。

设 $f: X \rightarrow Y$ 。若 $A \subseteq X$ 且 $B \subseteq Y$, 则令

$$f(A) = \{y | y \in Y \text{ 且有 } x \in A \text{ 使 } y = f(x)\}$$

$$f^{-1}(B) = \{x | x \in X \text{ 且有 } y \in B \text{ 使 } f(x) = y\}$$

称 $f(A)$ 为 A 在 f 下的象集, $f^{-1}(B)$ 为 B 在 f 下的原象集。并称 $f(X)$ 为 f 的值域, 常用 $\text{ran}(f)$ 表示。这时显然有

$$f(f^{-1}(B)) \subseteq B \quad \text{且} \quad A \subseteq f^{-1}(f(A))$$

而且当 $B \subseteq \text{ran}(f)$ 时, 还进一步有 $f(f^{-1}(B)) = B$ 。

设 $f: X \rightarrow Y, A_1, A_2 \subseteq X$ 且 $B_1, B_2 \subseteq Y$, 则有以下结论:

- (1) 若 $A_1 \subseteq A_2$, 则 $f(A_1) \subseteq f(A_2)$;
- (2) 若 $B_1 \subseteq B_2$, 则 $f^{-1}(B_1) \subseteq f^{-1}(B_2)$;
- (3) $f(A_1 \cup A_2) = f(A_1) \cup f(A_2)$;
- (4) $f(A_1 \cap A_2) \subseteq f(A_1) \cap f(A_2)$;
- (5) $f^{-1}(B_1 \cup B_2) = f^{-1}(B_1) \cup f^{-1}(B_2)$;
- (6) $f^{-1}(B_1 \cap B_2) = f^{-1}(B_1) \cap f^{-1}(B_2)$ 。

设 $f: X \rightarrow Y$, 若 $f(X) = Y$, 则称 f 为满射; 若当 $x_1, x_2 \in X$ 且 $x_1 \neq x_2$ 时恒有 $f(x_1) \neq f(x_2)$, 则称 f 为内射; 若 f 既是满射, 又是内射, 则称 f 为双射或一一映射。

如果 $f: X \rightarrow X$ 使得每个 $x \in X$ 皆有 $f(x) = x$, 则称 f 为 X 的恒等映射, 用 I_X 表示。

设 $f: X \rightarrow Y$ 且 $g: Y \rightarrow Z$, 这时可定义映射 $h: X \rightarrow Z$ 如下:

$$h(x) = g(f(x)) \quad x \in X$$

称 h 为 f 与 g 的合成映射, 用 $g \circ f$ 表示。

映射 $f: X \rightarrow Y$ 与映射 $f': X' \rightarrow Y'$ 相等, 是指 $X = X', Y = Y'$, 且对每个 $x \in X$ 皆有 $f(x) = f'(x)$ 。如果 $g: Y \rightarrow X$ 使 $g \circ f = I_X$ 且 $f \circ g = I_Y$, 则称 g 为 f 的逆映射, 并称 f 有逆映射。如果 f 有逆映射, 则其逆映射必是惟一的, 因此常用 f^{-1} 表示 f 的逆映射。

设 $f: A \rightarrow B, g: B \rightarrow C$ 且 $h: C \rightarrow D$, 则

$$(h \circ g) \circ f = h \circ (g \circ f)$$

$$f \circ I_A = f = I_B \circ f$$

$$f^{-1} \circ f = I_A \quad \text{且} \quad f \circ f^{-1} = I_B$$

$$(f^{-1})^{-1} = f$$

$$(g \circ f)^{-1} = f^{-1} \circ g^{-1}$$

而且 f 有逆映射当且仅当 f 为双射。 (王兵山)

yingcipan cunchuqi

硬磁盘存储器(hard disk storage) 使用硬磁盘作存储媒体, 以磁头为电磁转换器件进行数据记录的直接存取存储设备。它是旋转磁表面型存储设备的一种。硬磁盘存储器由硬磁盘驱动器、硬磁盘控制器、硬磁盘适配器和相关软件(驱动程序和管理程序)等部分组成。

硬磁盘驱动器的功能是驱动盘片按一定转速稳定旋转, 驱动载有磁头的存取臂到达且稳定在指定的半径位置上, 控制磁头在盘面磁层上按一定的记录格式和编码方式进行写入和读出(参见硬磁盘驱

动器)。

硬磁盘控制器是控制硬磁盘驱动器实施数据的存入与取出的设备。硬磁盘控制器和硬磁盘适配器是计算机主机和硬磁盘驱动器之间的连接部件,它按照主机发来的命令控制驱动器进行磁道的寻找(存取臂定位)、磁头的选接、数据的写入(存)与读出(取)以及数据的缓冲、串并转换、编码等操作(参见硬磁盘控制器和硬磁盘适配器)。

硬磁盘 一种硬质圆形磁表面存储媒体。它是构成硬磁盘存储器的重要部件。硬磁盘也称硬盘片。硬磁盘由盘基(基片)及附着在它两面上的底层、磁性记录层、保护层、润滑层等构成。它是一种需在特殊条件下经多道工序精心加工的精密制品。基片材料多年来一直以硬质铝镁合金为主。后来,在3.5英寸及其以下的小、微型硬磁盘驱动器中已开始使用玻璃材料作基片。此外,在开发中的还有玻璃釉陶瓷基片。磁性记录层有颗粒涂敷磁层和连续薄膜磁层两种类型。铁氧体磁层和含钴铁氧体磁层属颗粒涂敷磁层,从最初的磁盘开始一直沿用至今。铁氧体涂敷磁层的性能稳定,技术成熟,但因厚度不能太薄,记录密度受到限制。金属镀膜、金属溅射膜、铁氧体溅射膜等属连续薄膜磁层。金属溅射膜薄膜磁层可做到非常薄的均匀层,矫顽力高,适合高密度记录用。

硬磁盘存储器记录密度高,存储容量大,存取时间短,可作联机存储器使用;另外,可靠性好,单位存储容量的成本低。硬磁盘存储器在计算机辅助存储设备中占重要地位。

硬磁盘接口 连接硬磁盘与主机使之能互相协同工作的协议或标准。硬磁盘存储器从20世纪50年代诞生以来,容量不断扩大,数据传输速率不断提高,在存储器的结构上也有不少变化。硬磁盘的接口也从设备级接口到系统级接口,进而向智能化的接口方向发展。

设备级接口 是硬磁盘驱动器与硬磁盘控制器之间的接口。其结构特点是硬磁盘控制器作为主机系统的一部分,通过系统总线与主机通信。而硬磁盘驱动器作为一个外接设备接到控制器。设备级接口有SMD,ST 506/412,ESDI等。

(1) SMD接口 采用A,B两根电缆连接。A电缆用于控制器向驱动器发送命令和驱动器向控制器反馈状态信息。B电缆用于传送读写数据和时钟信号。SMD接口多用于大中型计算机。SMD接口的数据传输速率为1.5~3 MB/s。

(2) ST 506/412接口 用于小容量的5.25英寸和3.5英寸的硬磁盘驱动器。采用A,B两根电缆。A电缆用于传送控制信号,B电缆用于传送读写数据。由于传送的是未经分离的改进调频制(MFM)信号,易受干扰引起数据出错。数据传输速率为5 Mb/s(0.625 MB/s)。ST 506/412接口广泛应用于8位和16位微型计算机上。在ST 506/412的基础上作了改进的ST 506/412HP接口,数据传输速率达15 Mb/s。

(3) ESDI接口 将数据的编码和译码放在驱动器内完成,在电缆上传送的是不归零制信号,提高了抗干扰能力,数据传输速率提高到15 Mb/s。

设备级接口现已很少采用。

系统级接口 主机系统与硬磁盘子系统之间的接口。由于控制器与驱动器作成一体,形成一个硬磁盘子系统。接口的一端是主机系统中的适配器,另一端是硬磁盘子系统内的控制器。常见的系统级接口有IDE接口、EIDE接口、SCSI接口等。

(1) EIDE接口 是广泛应用的硬磁盘接口。它是IDE接口的发展。特点是将控制器集成到驱动器中,并备有高速缓存,以提高存取速度。EIDE有两种数据传输模式,即PIO模式和DMA模式。DMA模式的数据传输速率高。EIDE接口只用一根电缆与主机系统相连。由于EIDE接口规范与EISA总线基本上一致,适配器的结构就非常简单,只有总线缓冲、地址译码等电路。通常采用一片集成电路直接装在主机板上。

EIDE接口还可以连接符合ATAPI标准的磁带机和光盘机。

(2) SCSI接口 具有智能的多任务输入输出接口。不但适用于硬磁盘,还用于其他有SCSI接口的外围设备,如扫描仪、光盘驱动器、打印机、磁带机等。可通过连接在SCSI总线上的SCSI控制器管理光盘等其他外围设备。在PC机上使用SCSI接口的外围设备时,需要加适配器。SCSI适配器有多种类型,如ISA,EISA,VESA,PCI等,分别适用于各种微型计算机总线。

SCSI接口有许多类型,比如SCSI-1,SCSI-2,Fast SCSI,Wide SCSI等,这些接口的总线宽度和数据传输速率不同。SCSI接口在不断发展中(参见外存储设备接口)。

参考文献

吴产乐主编. 微机系统与接口技术. 武汉:华中科技大学出版社,2002
(刘锡刚)

yingcipan kongzhiqi

硬磁盘控制器 (hard disk controller) 连接计算机与硬磁盘驱动器,以实现计算机对硬磁盘驱动器进行数据存取的控制部件。它将计算机需要存储的数据经缓冲、并串转换、编码等处理后送往硬磁盘中由计算机程序指定的地址处进行记录,或从硬磁盘中取出计算机需用的数据,经译码、串并转换、缓冲等处理后送往计算机主存,并及时检测控制器中的状态信息,供计算机处理。

硬磁盘控制器的基本结构视计算机系统及其对硬磁盘驱动器配置的要求而异。在大中型计算机系统中,一般将其设计成位于计算机通道与硬磁盘驱动器之间的独立部件,但它在功能和结构上,可能与两者有所交叉。有的控制器承担了部分或大部分原来由通道完成的功能,这就构成了各种结合型控制器。在硬磁盘驱动器一侧,控制器一般通过菊花链式或辐射式电缆直接与若干台硬磁盘驱动器相连。但也有的系统在控制器与硬磁盘驱动器之间,设置若干驱动控制器,每个驱动控制器管理若干台硬磁盘驱动器,而将一些公共环节如编译码器等集中于控制器。这种结构可增加系统中硬磁盘驱动器的个数,并利于实现硬磁盘驱动器的并行操作。在小型和微型计算机中,一般配置 2 至 4 台小型硬磁盘驱动器。小型机的控制器有的通过总线适配器与高速总线相连。微型计算机的控制器通常是一块插卡,插于系统总线插槽上。微型计算机的硬磁盘控制器又常称为**硬磁盘适配器**。

按某种硬磁盘接口设计的硬磁盘控制器只能配接具有相同接口标准的硬磁盘驱动器。不同的接口标准影响到硬磁盘控制器与硬磁盘驱动器之间功能部件的界面划分,从而影响硬磁盘控制器的结构。ST 506/412 接口传送的读写数据是按某种调制码编码的脉冲串,其编码和译码电路(数据分离器)放在控制器上,控制信号是单线定义的,每根线具有单一的功能。ESDI 和 SMD 接口传送的数据是不归零制(NRZ)串行数据,编码电路和数据分离电路则放在驱动器上。但 ESDI 的控制信号可用软件设置为步进方式和串行方式。步进方式的信号定义与 ST 506/412 相同,串行方式则用 16 位串行数据来传送命令和配置信息,通过其中的若干位串行数据的编码来识别不同的命令或配置和状态信息。SMD 的命令和状态则是接口数据线上的并行编码数据。SCSI 和 IDE 一类系统级接口已将控制器的全部功能都放在驱动器上,控制器与驱动器合而为一,形成

所谓“**嵌入式控制器**”。这种具有系统级接口的驱动器,通过主机适配器与计算机直接相连。SCSI 还有另一种配置:它仍具有独立的控制器,该控制器连接具有设备级接口的驱动器,例如 ESDI 驱动器,再设计一个针对具体主机的具有 SCSI 接口的主机适配器,与该控制器连接。不同级别的 IPI 及其他接口对控制器的要求各不相同。可见,驱动器的接口标准不同,控制器的结构和设计就有差别。

硬磁盘控制器由主机接口逻辑、控制逻辑和驱动器接口逻辑三部分组成。主机接口逻辑包括若干个寄存器和与主机接口有关的控制逻辑,用来控制通道或总线与控制器之间的命令、数据、状态等信息的传送。控制逻辑通常由以微处理器或微控制器为控制核心的控制电路及并串、串并转换电路、检纠错编译码电路等组成,ST 506/412 接口则还包括调制码编译码电路。控制逻辑的主要任务是进行命令译码,产生各种微命令,并变换为符合驱动器接口要求的序列控制信号,同时控制读、写信号的变换、处理与缓冲,也使之符合接口规约的传输要求,在序列控制信号的控制下,一步步完成程序所要求的操作。

硬磁盘控制器所执行的命令大致可分为以下 5 类。

(1) **控制型命令** 用于执行不涉及在磁盘上记录或读取信息的一类操作,其中主要是查找命令,它用于控制磁头定位于所需读、写的磁道上。

(2) **读命令** 主要用于读出磁盘上指定记录区段的串行数据,经变换和处理后,并行送到计算机主存。

(3) **写命令** 主要用于将计算机送来的待记录的并行数据,经变换和处理后记录到磁盘上指定的记录区段内。这样,原来的记录就被改写了。数据是按规定的磁盘格式记录在磁盘上的,因此,写命令还包括格式化写命令。格式化写命令用来将各区段写上规定的格式。可按硬分段或软分段将一条磁道分为若干个扇区,用一个索引脉冲表示一条磁道的起点,硬分段用一个扇区脉冲表示一个扇区的开始,软分段的扇区则由识别码来表示。每个扇区分两个记录子段。一个为识别段,该子段记录了用于读出同步的同步信息,表明该子段为识别段的识别地址标记,以及该扇区所在的磁道号、扇区号、磁头号,扇区记录的字节数以及检验码。另一子段为数据段,它记录有同步信息,表明该子段为用于记录用户数据的数据地址标记,以及用户数据和检纠错编码。子段末尾还留有间隙,其上记录规定的数码,以便匹

配盘速变化及电路延时。

(4) 断定型命令 用于检查错误状态的具体原因,以便计算机据此进行处理。

(5) 搜索命令 用于将从计算机送来的信息与从磁盘读出的信息进行比较,当搜索条件满足时,将相应的标志位置位,供计算机程序使用。

硬磁盘控制器应保证能按计算机程序指定的地址存取数据,其工作原理如下。

从硬磁盘读数据到计算机主存的操作过程是:首先向硬磁盘驱动器发送一个命令,指出要读的磁道和扇区,启动寻道操作,将磁头定位于目标磁道上;然后搜索目标扇区,磁头扫描到识别段的地址标记后,将其后面的磁道号、扇区号、磁头号等地址信息与计算机要求的地址信息相比较;一旦找到目标扇区,即读出其后的数据段中地址标记后的数据。由磁头读出的被记录的磁化状态翻转信号,经过驱动器放大和处理后,还原成对应于磁介质上磁通反转时刻的脉冲信号。这是一个时钟和数据的复合信号,与某种调制码的编码相对应。数据分离电路利用锁相环及译码电路对这个信号进行同步和译码,以便跟踪由于磁盘主轴转速变化引起的编码脉冲的平均频率变化,产生出正确的数据窗口和时钟窗口,减少读出的错误,分离出数据和时钟。电路噪声、串扰、磁头定位偏差、介质缺陷等多种原因,都会引起读出的错误,其中介质缺陷是永久性出错的原因。非永久性出错可用命令重执来克服,永久性出错则需借助于纠错码(ECC)来保证数据可靠的存储和恢复,这就需要检纠错编译码电路或相应软件。串行的数据编码脉冲被同步和译码为 NRZ 数据后,经过 ECC 电路检错和纠错,再将串行数据变换为并行数据,送入缓冲存储器中。最后,通过直接存储器存取(DMA)传送给主机。

数据写入磁盘是上述读过程的逆过程。在磁头定位到所需磁道并搜索到所需记录的扇区后,就启动写操作。待写数据先通过 DMA 从主机送到控制器的缓冲存储器中,并通过并串转换电路进一步将这个并行数据转换为串行数据,经过 ECC 编码电路进行检纠错编码,再经过调制码编码电路变为某种格式的编码数据。在将这个编码数据送到硬磁盘驱动器写电路进行写入之前,还要经过写预补偿电路处理,以克服脉冲拥挤效应引起的峰点漂移。位漂移超过一定的容差,就会产生读出错误,特别在磁盘内圈磁道上影响更为显著。现在大都采用写前补偿的方法来解决这个问题。这个方法是根据记录数据

的模式,预测某一位发生漂移的方向,写入时适当调整写入的时刻,来补偿这个漂移,这由写预补偿电路来完成。

写格式化操作和常规的写操作相类似,只要按序提供格式化所需的数据即可。这可由一种称为格式器的电路来完成,也可用软件提供格式化数据。

早期的硬磁盘控制器用逻辑电路或微程序控制电路构成,芯片集成度低,体积大。后来出现了多种控制器专用芯片,它们集成了部分或大部分控制器的功能。有的专为某种驱动器接口而设计;有的可适用于各种接口;有的集成了控制数据通路所需的全部逻辑,而将其他控制逻辑由其他电路(如微控制器等)处理,使控制器的设计可灵活适应某种计算机系统或某种接口的硬磁盘驱动器。专用芯片及微控制器的应用,使硬磁盘控制器智能化。现在,硬磁盘控制器仅是一块芯片,放在主机板或硬盘机上。

参考文献

刘乐善,叶济忠,胡盛斌.微机接口技术及应用.武汉:华中理工大学出版社,1993 (叶济忠)

yingcipan qudongqi

硬磁盘驱动器(hard disk drive, HDD) 驱动固定安装在主轴上的硬磁盘并实施数据存取的设备。它是目前使用最广泛的,也最能反映磁盘存储技术水平的一种驱动器。不同盘片尺寸、不同记录密度、不同容量档次和速度性能档次的硬磁盘驱动器分别适应从大型计算机到微型计算机的各种不同需要。以前,硬磁盘驱动器分为大型和小型两类。后来,为适应微型计算机的迅速发展,又涌现出一系列在外形尺寸上与之匹配的超小型、超薄型的硬磁盘驱动器。这类设备有时称为微型驱动器,以便区别于小型驱动器。由于硬磁盘驱动器都应用了温切斯特技术,故又称温切斯特盘驱动器、小型温盘驱动器、微型温盘驱动器等。

第一台磁盘驱动器是 1956 年完成的 IBM 360 (RAMAC)。它的容量是 5 MB,位密度是 3.9 b/mm (100 bpi),道密度是 0.8 道/mm (20 tpi)。采用固定式盘片,盘片直径是 610 mm (24 in),一共用了 50 片。整个驱动器是一个庞然大物。

在 1963 年以前的磁盘存储技术的初始阶段,所有驱动器都采用固定盘片的形式,显然容量不能满足使用的要求。因此提出可换盘组,作为存储媒体的盘组可脱离驱动器保存,理论上可做到存储容量无限大。

第一台实用的可换盘驱动器是 1966 年完成的 IBM 2314。它的容量是 29.2 MB, 位密度是 86 b/mm (2 197 bpi), 道密度是 3.9 道/mm (99 tpi), 采用的盘片直径是 356 mm (14 in)。

1963 年至 1975 年的十多年间, 各种可换盘驱动器盛行一时。为了确保存储媒体的广泛互换性, 盘片、磁头、磁道的几何尺寸、记录密度和记录格式等的物理特性和电磁特性均有国家或国际标准。

1976 年产的 IBM 3350 型固定硬磁盘驱动器问世之后, 由于存储容量的显著增长, 脱机扩大容量的方法已不再必要, 驱动器又以盘片固定的形式发展。盘片固定更利于记录密度和可靠性的提高。可换盘组最大也是最后的容量是 200 MB。

IBM 3350 型磁盘驱动器是在初次采用温切斯特技术的 3340 型驱动器的基础上经过改进而发展起来的。温切斯特技术的基本内容如下:

(1) 磁头、磁盘片、主轴及装载头臂的小车等机械公差要求严格的关键零部件装在一密封的壳体内, 此部件称作头盘组合件 (HDA);

(2) 低质量、小尺寸、轻浮力的平轨磁头浮动块, 可与盘片作接触启停, 盘片磁层表面涂有润滑剂, 盘片不转时头与盘接触, 当盘达到一定转速时浮动块浮起, 当盘速降至一定值时浮动块经一段滑行后降落到盘面上;

(3) 读前置放大电路、写电流驱动电路及选头电子开关均集成在一个芯片上, 此芯片安装在各磁头存取臂上以便尽量减少外界电磁场对磁头引线的影响。

采用温切斯特技术时, HDA 安装在驱动器的减震架上, 需和驱动器连接的部位仅限于主轴电机、存取臂驱动音圈电机、净化空气输入口及读写电路插件等处。由于磁头与磁盘组合成为一个整体, 磁头与磁盘的相对关系是固定的, 磁头所读出的乃是由它本身写入的信息, 因此无需像可换盘硬磁盘驱动器那样为保证盘片可换性而考虑严格的尺寸公差要求, 从而使密度得以大幅度提高。由于 HDA 的密封结构, 驱动器可靠性也有显著提高。轻浮力小型浮动块可以在较小的头盘间隙飞行, 加上新型磁层, 位记录密度也较以往可换型有较大提高。以 IBM 3350 型驱动器的推出为转折点, 固定硬磁盘驱动器一直持续向前发展。从 1976 年的 IBM 3350, 到 1993 年的 IBM 3390-9, 在不足 20 年的时间里, 道密度由 18.9 道/mm 提高到 132 道/mm, 位密度由 253 b/mm 提高到 3300 b/mm, 单机容量由 0.63 GB

提高到 102 GB。

3350 及其兼容或同类型产品都采用 356 mm (14 in) 或 203 mm (8 in) 直径的盘片, 整机体积太大, 这显然不适合在微型计算机中应用。小型温盘驱动器是利用了大型驱动器成熟的温彻斯特技术根据小型的特点发展起来的。1980 年出现的 ST 506, 采用直径为 5.25 in 的盘片, 虽然容量只有 6 MB, 但体积大小很适合微型计算机使用。ST 506 的平均寻道时间为 85 ms, 面密度为 1.96 Mb/in²。后来, 陆续在小微型硬磁盘驱动器中广泛应用了金属薄膜磁层、薄膜磁头或隙含金属磁头 (MIG)、微型浮动块、数字伺服以及区位记录 (ZBR) 技术, 还采用了部分响应最大似然 (PRML) 信号处理技术和磁变阻 (MR) 磁头等新技术, 从而在道密度、位密度、存储容量以及数据传输速率方面都有大幅度的提高。以 2003 年出产的 Barracuda-50 为例, 容量为 50.1 GB, 比 ST 506 提高了 104 倍, 平均寻道时间为 7.4 ms, 提高了近 10 倍, 面密度为 325.2 Mb/in², 提高了约 160 倍。

20 世纪 80 年代以来, 盘片直径从早期的 5 英寸向更小的尺寸推移。2003 年, 3.5 英寸是主流产品。单片容量已达 80 GB, 单台容量达到 300 GB, 转速高达 15 000 r/min。2.5 英寸、1.8 英寸和 1 英寸以下的超小型产品也已应用。

硬磁盘驱动器的发展趋势已较明朗, 微米级超窄磁道的磁变阻 (MR 及 GMR) 头、高矫顽力低噪声薄膜磁层、垂直磁记录、0.1 μm 级的飞行高度技术、PRML 信号处理技术等可以推动驱动器的记录密度和存储容量继续大幅度增长。

参考文献

1. Tsang C, et al. Gigabit Density Recording Performance Study of Dual-Element MR/Inductive Heads on Thin Film Disks. IEEE Trans. Mag., Sept., 1990, 26(5): 1 689 ~ 1 693
2. Yogi T, Tsang C, Nguyen T A, et al. Longitudinal Media for 1 Gb/in² Areal Density. IEEE Trans. Mag., Sept., 1990, 26(5): 2 271 ~ 2 276
3. Futamoto M, et al. Investigation of 2 Gb/in² Magnetic Recording at a Track Density of 17 ktpi. IEEE Trans. Mag., Nov., 1991, 27: 5 280 ~ 5 285

(刘锡刚)

yingcipan qudongqi ceshi

硬磁盘驱动器测试 (hard disk drive testing)

通过测试仪对硬磁盘驱动器机械、电气性能进行

的检测和评估。硬磁盘驱动器应在研制、生产、维修等过程中,进行严格的检测和评估(对于可换盘硬磁盘驱动器还应检测其互换性),以保证其质量。

硬磁盘驱动器机械与电路的测试和调整内容主要有:索引信号,零道信号,准备好信号,主轴转速,寻道,磁头定位(径向位置及方位角),磁头分辨率,磁头读出信号幅度,磁头写入波形,数据信号抖动性,内、中、外磁道信号的不对称度(测信号漂移),内、中、外磁道读裕量(测误码率),连续写读,格式化磁盘。

硬磁盘驱动器测试仪大致可分练习仪和测试分析仪两大类。

练习仪 是一种低档的非智能或具有一定编程功能的简易型检测仪。一般用于完成对硬磁盘驱动器的索引信号、零道信号、准备好信号、磁头寻道、写读、选头等功能测试,与示波器联用时还可观测驱动器读写数据信号。检测和调整可换盘硬磁盘驱动器时,还要用到调整盘。练习仪结构简单,轻巧,使用方便,一般供维修使用,每次仪可测试一台驱动器。

测试分析仪 是一类中高档功能检测仪,采用微处理机或计算机控制,可自动操作,也可手动操作。除了练习仪的基本功能外,还能对硬磁盘驱动器进行读信号、相位裕量、数据分离、信号不对称度等指标的定量分析,可用来对硬磁盘驱动器进行全面的检测和评估。这类仪器能精确地进行出错定位和记录,并可同时或顺序测试多到 64 台驱动器。测试程序、测量值及测试结果可在显示屏上实时显示或同时由打印机打印输出。仪器的结构型式有便携式、台式、落地式等。

参考文献

Greenwood D. Disk-Drive Testers. Electronics Test, 1984(2): 68~81
(周学仁)

yingcipan shipeiqi

硬磁盘适配器(hard disk adapter) 连接微型计算机总线与硬磁盘驱动器,以实现微型计算机对硬磁盘驱动器进行数据存取的接口部件。它既起信号适配的作用,又起控制的作用。前者指缓冲与匹配微型计算机总线与硬磁盘驱动器间的传输速率,并将微型计算机总线传送的信号转换为符合驱动器接口规约的信号。后者指执行硬磁盘输入输出控制程序,将计算机需要存储的数据经缓冲、并串转换和编码等处理后,送往硬磁盘中由程序指

定的地址处进行记录;或从硬磁盘取出计算机需用的数据经译码、串并转换和缓冲等处理后,送往微型计算机主存;并及时检测适配器中的状态信息,供微型计算机处理。故硬磁盘适配器又常称为硬磁盘控制器。

硬磁盘适配器和大中型等计算机的**硬磁盘控制器**的基本功能、基本工作原理和寻道、读、写等基本控制过程是相同的;但由于它们所要求的硬磁盘驱动器容量、数量、数据传输速率等差别较大,因而它们之间在具体结构、功能强弱以及驱动软件的复杂程度等方面都有一定差异。大中型计算机的硬磁盘子系统一般是独立于主机之外的设备,规模较大;一些小型机的硬磁盘子系统通过专用的适配器与高速同步总线相连;而在微型计算机中,硬磁盘适配器一般为一块插卡置于主机箱内,用以控制一至两台 130 mm 及以下的小盘径硬磁盘驱动器,或和软磁盘适配器合做在一起,近年来还和串、并接口一起做在一块多功能卡上。

硬磁盘适配器由主机接口、控制逻辑、驱动器接口三部分电路组成。主机接口设置若干个寄存器及中断等逻辑电路,控制程序通过相应端口对寄存器寻址,进行读取状态、传送命令和数据等操作。控制逻辑包括串并、并串转换,调制码和检纠错码编译码电路等,大多以微处理器为控制核心,对数据进行处理,并根据命令性质和有关参数,形成寻道、读、写等符合驱动器接口规约的控制信号序列,通过发送与接收驱动电路与硬磁盘驱动器相连。目前,很多小型硬磁盘驱动器都采用 SCSI 等系统级接口,适配器的控制功能已融于硬磁盘驱动器内,通过匹配主机总线和系统级接口的主机适配器与这种硬磁盘驱动器连接,硬磁盘适配器仅是一块简单的芯片。

参考文献

刘乐善,叶济忠,胡盛斌.微机接口技术及应用.武汉:华中理工大学出版社,1993 (叶济忠)

yingjian miaoshu yuyan

硬件描述语言(hardware description language, HDL) 描述电子系统硬件结构与功能行为的语言。硬件描述语言能用于表达元件(门级、芯片级、插件及系统级)的硬件设计。同时也是一种能提供统一的设计数据格式和信息交换标准的工具。应用 HDL 语言,能够进行逻辑设计、逻辑综合、模拟验证、测试、诊断和修改。

硬件描述语言早在 20 世纪 60 年代就开始研

究,如 1965 年发表的 CDL,1968 年发表的 DDL,1973 年发表的 AHPL 等都是重要的硬件设计语言。这些语言之间很少有实质性的差别,也没有得到统一和推广,因而妨碍了它们的发展。超高速集成电路硬件描述语言 VHDL 语言是一种标准化的硬件描述语言。1980 年美国国防部提出超高速集成电路 VHSIC 发展计划,要求各承担超大规模集成电路产品订货的公司都要以新设计的硬件描述语言 VHDL 作为统一的设计数据文件格式和信息交换工具。对所设计生产的集成电路,必须用 VHDL 描述,并用它的模拟系统模拟验证。美国国防部这种强制性的要求,有效地促进标准化工作。1986 年美国电气与电子工程师协会 IEEE 以 VHDL7.2 版本为基础制定了标准的 VHDL 版。1987 年成为美国国防部标准,并编号为 IEEE Std 1076。尔后不到 3 年时间,美国、日本和欧洲原有的和新研制的电子设计自动化系统,全部采用 VHDL 作为描述语言。1993 年推出新标准。1999 年推出美军扩展标准 VHDL-AMS extension。

VHDL 语言的特点是:

(1) 能描述硬件的构成,即微电子器件和元件系统的功能构成;

(2) 能描述硬件的行为,即微电子器件和元件系统如何工作,包括数据转换、定时、电信号的变化等;

(3) 能描述设计实体,说明元件的外部特性,以便组装引用;

(4) 能描述结构的连接、组装关系,以便构成一个复杂的系统;

(5) 具有丰富的类型,除具有数值类型、字符类型、枚举类型外,还有长度、时间、电容量、阻值等物理量类型;

(6) 具有丰富的表达能力,除提供一般的说明、算术、逻辑运算语句外,还提供并发、进程等控制语句和接口语句。

VHDL 语言在逻辑设计、逻辑模拟、故障测试诊断、电路建模、逻辑综合和验证及行为综合等有广泛的应用。进一步的发展是增加自诊断系统和自修改系统功能。

参考文献

1. 刘明业. 数字系统设计自动化. 北京: 电子工业出版社, 1994

2. 王志华等. 数字集成系统的结构化设计与高层次综合. 北京: 清华大学出版社, 2000

(潘雪增)

yingjian tongbu jizhi

硬件同步机制 (hardware synchronization mechanism)

在并行处理系统中,通过采用硬件实现低层或原语级控制信息的通信,以保证多个进程具有正确执行顺序的机制。同步的目的是保证并行处理结果的正确性,它可以通过硬件、固件或软件(用户程序或操作系统)实现,硬件同步机制是任务层软件同步的基础。同步是获得并行加速所付出的主要开销,有效的硬件同步机制是实现并行处理的一项关键技术。

在单处理机中,指令的顺序由程序计数器确定。早期的计算机未提供不可中断的读改写指令,实现同步非常困难。E. W. Dijkstra 早在 20 世纪 60 年代就提出了只用 ALGOL 60 语言的标准语句实现同步的方案,他的方案中实际上隐含了多处理机必须保持顺序一致性这一假定,即并行处理系统中所有处理机观察到的各处理机的存取操作都是按同一次序进行的。由于存取数据可以在寄存器、高速缓存、主存储器以及磁盘中进行,存取时间相差很大,先执行的存取指令可能后完成。从别的处理机观察一个处理机,观察到的存取次序可能与这个处理机上串行编程规定的次序是不一致的。如果要求多处理机系统运行时严格遵守顺序一致性,就可能导致系统性能的明显下降。对于当今的多处理机系统,E. W. Dijkstra 的同步方案已不重要,因为功能不断增强的不同形式的读改写硬件同步原语已逐渐被采用,专门的同步指令以及比顺序一致性要求低的各种一致性协议可以保证并行计算结果的正确性。

硬件同步机制的基础是不可中断的原子操作,所谓原子操作是指执行读改写方式的同步指令时,必须等指令执行完毕,别的指令才能执行,执行过程不允许中断。为了防止同时修改一个共享变量,保证数据的完整性与顺序地进入临界区,同步原语必须具有原子性。所谓临界区是指必须顺序执行的一段程序,每个临界区由一个同步变量即信号量保证每一时刻最多只有一个进程进入。已被采用的硬件同步原语或指令有好几种,其中具有代表性的包括测试与设置,增 1 与减 1,比较与交换,取与加等。

(1) 测试与设置(Test and Set)是最简单也是最早采用的硬件同步原语。使用这一原语要求每一共享数据附加一位专门的同步变量,或称信号量。存取共享变量时先要执行一条不可中断的读改写方式的指令 Test and Set。第一个通过 Test and Set 同步

原语发现信号量为 0 的进程被允许存取共享变量,在执行 Test and Set 原语时已将信号量置 1。因此,在一个进程存取共享变量时,其他进程不可能再存取共享变量,从而保证了互斥存取。当存取结束时再将同步变量置 0,允许其他进程通过竞争去存取共享变量。

Full/Empty 同步原语与 Test and Set 类似,只是增加了两个两位字段 SAC 和 DAC,分别控制同步读与同步写,而且共享变量与信号量可同时存取,以节省存取时间。这一机制也已用于数据流计算机,以保证正确的读写次序。

(2) 增 1 与减 1 (Increment and Decrement) 可以看成是 Test and Set 原语的扩展。考虑一个长度为 M 的共享缓冲器,即缓冲器含有 M 个共享的存储单元。初始时共享缓冲器的信号量置为 M 。只要不同进程访问缓冲器中的不同单元,最多 M 个进程可同时访问缓冲器。每一进程进入临界区,信号量减 1,完成临界区操作则加 1。Test and Set 原语相当于缓冲器长度为 1 的情形,每一时刻最多只允许 1 个进程进入临界区。Increment and Decrement 原语比 Test and Set 原语效率高,但如果处理不当,可能出现死锁,即当大量进程同时要求访问缓冲器时,可能使信号量小于 0,使得一段时间内无法对缓冲器进行存取。

(3) 比较与交换 (Compare and Swap) 硬件同步原语将临界区缩小到只有 1 条比较与交换指令,这明显地降低了同步开销。执行 Compare and Swap 需要两个寄存器,一个存共享变量旧值,另一个存其新值。计算共享变量新值时不能进入临界区,当新值算出结果以后,再执行不可中断的 Compare and Swap 指令,以测试共享变量的值是否改变,如未改变,则赋以新值,如已改变,则将共享变量当前值存于旧值寄存器。与前两种同步原语不同,在执行 Compare and Swap 之前,允许多个处理机修改共享变量。队列指针是最常用的共享变量,Compare and Swap 原语提供一种途径,允许多个处理机同时修改队列指针而只花费很小的开销。Compare and Swap 是一种很有效的同步原语,但正确地使用这一原语较困难。

(4) 取与加 (Fetch and Add) 是一种完全并行、不需要临界区的同步原语,多台处理机可同时执行 Fetch and Add 指令。执行 Fetch and Add 指令要求互连网络具有存储和整数加法功能。当多个处理机通过 Fetch and Add(V, e) 指令存取同一共享变量 V

时,通过合并网络的开关模块,将存取及修改(即增量 e_i)要求合并成一个包括总的增量 e 的存取要求。不管有多少个同时存取的要求,只需对共享变量读改写 1 次。从存储器中取到的结果再通过网络开关按每个处理机不同的修改要求送到各处理机。Fetch and Add 同步原语取消了必须顺序执行的临界区,大大降低了同步开销,为构造例如 1 000 到 10 000 个处理机组成的大规模并行处理系统提供了一种高性能的同步机制,但由于附加的硬件成本太高,编程也十分困难,不适于中小规模的并行处理系统。

在消息传递类型的并行计算机中,消息的发送与接收起到与共享存储计算机中同步原语一样的同步作用,对消息传递同步机制最基本的硬件支持是连接发送与接收处理机的通信通道。消息传递的同步原语非常简单,可以用软件通信协议或专门的硬件实现。通信可以是同步方式,也可以采用准异步方式或完全异步方式。在完全异步方式的消息传递中,接收方不发出任何请求,发送方只要数据准备好就发送,不管接收方当时是否需要。这种同步原语对程序员完全透明,编程非常方便,但接收方必须有足够容量的匹配单元随时接收发来的消息。

设计低成本高性能的硬件机制支持并行计算机的同步一直是计算机追求的目标,采用硬件实现的屏障是受到广泛注意的同步技术。采用加法器、与门、线或等简单器件可以实现屏障功能。对于包含上千个处理机的并行处理系统可以采用总线连接若干专门的屏障芯片实现同步。数据流计算机的每一条指令都要同步,高效率的同步机制显得更为重要,需要特殊设计的匹配单元实现同步。

实现高速缓冲存储器一致性也能起到同步的作用,对于规模较小的系统,用基于总线的高速缓冲存储器一致性协议实现同步可能是合适的选择,但对于包含数百个处理机以上的并行处理系统,专门的同步子系统可能更合算。比如, Cray T3D 系统实现了能够支持数千个处理机的“屏障”和“找到”等同步操作的专用硬件子系统。并行处理系统的同步性能可用每秒执行多少百万条同步指令来表示,记为 MSYPS。一般来讲, MSYPS 远小于 MIPS (百万条指令每秒)。增加处理机数量有可能提高系统的 MIPS,但不能增加 MSYPS。统计表示,平均每执行 50 条左右的指令就要执行 1 条同步指令,因此, MSYPS 是影响并行处理系统性能最主要的因素,应当受到重视。

参考文献

1. Stone H S. High Performance Computer Architecture. Reading, Massachusetts: Addison Wesley, 1993

2. 黄铠,徐志伟著.可扩展并行计算:技术、结构与编程.陆鑫达、曾国荪、邓倩妮等译.北京:机械工业出版社,2000 (李国杰)

yingjian yanzheng

硬件验证 (hardware verification) 检验所设计的系统是否满足规定技术指标的技术。具体讲,就是核对系统实现描述与技术指标描述的一致性。为此需给出两个描述的形式模型,要有两个描述之间一致性关系的形式概念,以及检验或证明一致性关系的某种方法。

可采用 VHDL 一类的硬件描述语言或采用包含硬件和软件子系统的“广谱”系统描述语言对系统进行描述。系统描述的基础是某种形式的代数或逻辑的形式模型。常用的模型有用于组合逻辑的布尔或命题逻辑,用于时序逻辑的有限自动机及状态图。系统技术指标描述 S 和系统实现描述 I 之间的一致性关系可以有:相等($S=I$),等价($S\leftrightarrow I$),逻辑蕴含($I\Rightarrow S$),条件逻辑蕴含($C(\text{input})\Leftrightarrow(I\Rightarrow S)$),其中 $C(\text{input})$ 为系统的运行环境)等几种。检验一致性所用的技术与所采用的形式模型密切相关。经常利用某些逻辑的特殊性质,开发专门的数据结构和算法,以支持该逻辑所需操作。比较有代表性的是布尔表达式的图形技术指标(规范)表示法(称之为有类型有序二进制判定图),它对实现布尔运算非常有效,可用于验证两个布尔表达式和两个确定性莫尔机器之间的等价性。此外,还有 μ 演算与图验证法等方法。 μ 演算可用来描述各种验证方法的总框架。图验证法包括时序逻辑模型验证、形式语言内涵、跟踪理论和通信系统计算。

硬件验证已用于某些领域,如 10 万个晶体管的组合电路, 2^{20} 个状态量级的控制电路。有些功能很强但自动化程度较低的技术,如基于定理证明器的交互技术已用来验证简单微处理器的某些功能行为。从理论上讲,对于任意电路,完全自动化的验证通常是不可能的。有时硬件验证虽然可做,但由于花费太大,实际上是不现实的。如果硬件验证能降低成本,则将对缩短新产品开发周期,提高产品可靠性等十分有用。对于关系人类生命安全的医药、民

航电子和核反应堆等系统,对于维修非常昂贵的管道和某些嵌入式系统,以及航天探测器等不可维修系统,硬件验证技术尤为重要。(刘恩德)

yinglianxian kongzhiqi

硬连线控制器 (hard-wired control unit)

由基本逻辑电路组成,对指令中的操作码进行译码,并产生相应的时序控制信号的部件。又称组合逻辑控制器。

硬连线控制器由指令部件、地址部件、时序部件、操作控制部件和中断控制部件等组成(参见“中央处理器”条目中的控制器部分)。其中操作控制部件用来产生各种操作控制命令,它根据指令要求和指令流程,按照一定顺序发出各种控制命令。操作控制部件的输入信号有:指令译码器的输出、时序信号和运算结果标志状态信号等。设计时根据指令流程、操作时间表得到各种操作控制命令的逻辑表达式,可采用基本逻辑电路与门、或门、与非门等组成的逻辑网络来实现。也可采用可编程逻辑器件 PLD 来实现。PLD 的“与”阵列及“或”阵列和操作控制命令的“与-或”逻辑表达式相对应,为设计组合逻辑控制器提供了一种理想器件。80 年代出现的通用阵列逻辑电路 GAL 与 PAL(参见**专用逻辑集成电路**)具有类似的结构,它不但可编程并且是可擦除的,提供了更大的灵活性。

组合逻辑控制器的最大优点是速度快。但因其线路复杂而且不规整,不便于调试、维护、修改,也不便于仿真不同的机器指令集。

参考文献

金兰. 计算机组织与结构. 北京:高等教育出版社,1986 (谢树煜)

yonghu jiemian

用户界面 (user interface) 计算机系统中实现用户与计算机通信的软硬件设施。又称用户接口,或人机界面。

用户界面的硬件部分包括用户向计算机输入数据或命令的输入装置及由计算机输出供用户观察或处理的输出装置。用户界面的软件部分包括用户与计算机相互通信的协议、约定、操纵命令及其处理软件。目前常用的输入输出装置有键盘、鼠标器、显示器、打印机等。常用的人机通信方法有命令语言、选项、表格填充及直接操纵等。

从计算机用户界面的发展过程来看,可分若干阶段:

(1) 控制面板式用户界面 这是计算机发展早期,用户通过控制台开关、板键或穿孔纸带向计算机送入命令或数据,而计算机通过指示灯及打印机输出运行情况或结果。这种界面的特点是人去适应现在看来是十分笨拙的计算机。

(2) 字符显示式用户界面 分时操作系统出现,使多个用户通过交互显示终端同时分享计算机资源。用户通过键盘输入字符型数据或命令,通过显示终端观察字符型的数据输出。这种界面的通信方法主要采用作业控制语言,后来发展为命令语言。其优点是功能强、灵活性好、屏幕开销少等。缺点是难学、难记、易错、需要一定键盘操作技能、显示不直观等。

(3) 图形用户界面 在 20 世纪 80 年代计算机图形技术基础上发展起来的图形用户界面,是当今计算机用户界面的主流。它采用 WIMP 技术,即窗口 (window)、图符 (icon)、选单 (menu) 及指点设备 (pointing device) 技术,采用多窗口系统为主要软件,以直接操纵为主要使用方法。其优点是操作简便、显示直观形象、适合初学者等。其缺点是对硬件资源要求较高、软件实现较复杂等。图形用户界面仍在不断改进发展中,如增加网络浏览功能、增强多媒体和三维功能、支持应用数据可视化、开发更好的界面构造工具和语言等。

(4) 新一代用户界面 虚拟现实技术将用户界面的发展推向一个新阶段:人将作为参与者,以自然的方式和计算机生成的虚拟环境进行通信。以用户为中心、自然、高效、高带宽、非精确、无地点限制等是新一代用户界面的特征。多媒体、多通道及智能化是新一代用户界面的技术支持。语音、自然语言、手势、头部跟踪、表情、视线跟踪等新的、更加自然的交互技术,将为用户提供更方便的输入技术。计算机将通过多感知通道来理解用户的意图,实现用户的要求;计算机不仅以二维屏幕向用户输出,而且以真实感(立体视觉、听觉、嗅觉、触觉……)的计算机仿真环境向用户提供真实的体验。新一代用户界面是计算机科学技术中最富有挑战性的领域之一。

参考文献

1. Helander M. Handbook of Human-Computer Interaction. Elsevier Science Pub. (North-Holland), 1988

2. Foley J D, VanDam A 著. 交互式计算机图形学基础. 唐泽圣,周嘉玉译. 北京:清华大学出版

社, 1986

3. 董士海. 计算机用户界面及其工具. 北京:科学出版社, 1994

4. 丁茂顺. 用户接口技术与交互系统构造方法. 北京:科学出版社, 1992 (董士海 蔡士杰)

yonghu jiemian guanli xitong

用户界面管理系统 (user interface management system, UIMS) 支持设计、构造、执行、评价、维护及管理计算机用户界面的软件。

随着人机交互系统的发展,用户界面已成为计算机系统的重要组成部分,它影响整个系统的可用性及使用效率。开发高质量用户界面需要很大工作量,因此人们希望把它从计算机系统中分离出来。以便使用户界面的开发不受系统功能核心的设计或改动的影响。这种界面与系统功能核心“分离”的思想促使了用户界面管理系统的诞生。W. Newman 在 1968 年建立的“反馈处理器”是最早的 UIMS。但 UIMS 作为专门术语是 D. J. Kasik 在 1982 年首次提出的。早期 UIMS 的功能仅限于原型构造或显示管理,可用性较差。1982 年召开的图形输入交互技术 (GIIT) 研讨会首次阐述了 UIMS 的概念、作用及结构模型等,此后 UIMS 有较快发展,出现了 TIGER, GWUIMS, Alberta UIMS 等实验系统。80 年代中后期出现了一些商品化 UIMS。我国也研制了一些实验性系统。

用户界面管理系统由两部分组成。其一是构造用户界面的工具集,称为用户界面开发环境或设计环境 (UIDE),其二是用户界面运行支持系统,称为用户界面系统 (UIS),它与应用系统的功能核心共同组成该应用系统。UIMS 框图如图 1 所示。

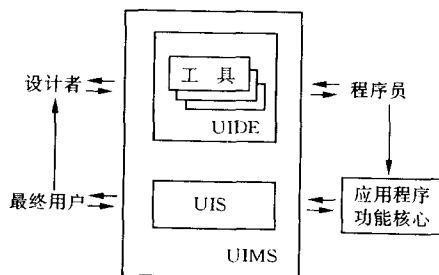


图 1 UIMS 框图

目前多数 UIMS 系统中 UIDE 的结构组成采用了 1983 年在 Seeheim 举行的用户界面管理系统国

际研讨会上提出的 **Seeheim** 模型 (如图 2 所示)。图中表示部件负责用户界面的外部表现,如屏幕管理、图形生成、设备管理、词法反馈及将输入数据转换成内部格式等。对话控制部件将表示部件的数据或请求,经检验传送给应用程序中合适的接口例程,它也将应用程序对请求的回答或其他数据传给表示部件。应用接口是从用户观点看到的应用程序的一种表述,它包括有关数据结构或对象的描述。用户

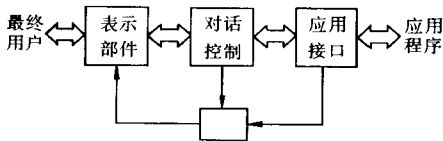
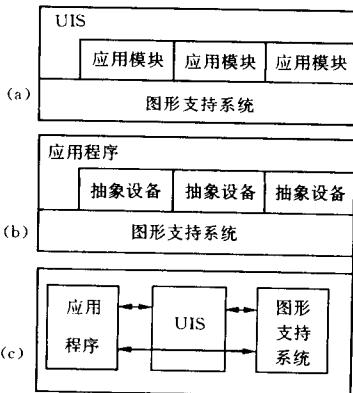


图2 Seeheim 结构模型

可使用的有关例程 (语义) 及其限制等。由于 Seeheim 模型只是一个“执行”模型,不反映整个界面软件生存期,且对于直接操纵方式所需的语义反馈很难适应,目前已提出许多改进。

UIMS 系统中 UIS 与应用系统功能核心的关系,涉及到用户界面的运行控制方式,它可分为三种类型:外部控制、内部控制及并行控制,分别如图 3 的 (a), (b), (c) 所示。外部控制是指用户通过 UIS 进行输入,并以此激发应用程序的各功能模块。内部控制是指应用程序根据输入请求的不同,来取得相应抽象设备的支持。并行控制是指应用程序功能核心与用户界面系统在同一层次上并行运行,它们间的通信协调通过并发进程来实现。

UIMS 的发展趋向是提高实用性,支持直接操纵方式,开发支持复杂用户界面 (如三维图形界面、多



(a) 外部控制; (b) 内部控制; (c) 并行控制

图3 界面的三种运行控制方式

媒体用户界面、分布式用户界面) 的设计技术,进一步重视人的因素研究等。尽管目前 UIMS 在可用性 & 功能方面还有不足,但它在提高界面开发效率及质量方面已显示极大潜力,期望它将会像现在的数据库管理系统一样广泛使用。

参考文献

1. 董士海. 计算机用户界面及其工具. 北京: 科学出版社, 1994
2. 程景云等. 人机界面设计与开发工具. 北京: 电子工业出版社, 1994
3. Pfaff G E. User Interface Management. Springer-Verlag, 1985 (董士海)

yonghu shujubao xieyi

用户数据报协议 (user datagram protocol, UDP) Internet 上的一种提供应用程序之间传送数据报的协议。和 TCP 一样, 它提供传输层的数据报服务。不同的是, 它提供不可靠的无连接数据报传送服务 (参见传输控制协议)。

TCP/IP 互联网能提供在主机之间传送数据的能力。每个数据报根据其目的主机的 IP 地址进行互联网中的路由选择。为了在给定的主机上能识别多个目的地址, 同时允许多个应用程序在同一台主机上工作并能独立地进行数据报的发送和接收, 在 TCP/IP 协议集中设计了用户数据报协议 (UDP)。

将每台机器看作是一些抽象的协议端口的集合, 通过协议端口能区分在一台机器上运行的多个程序。每个 UDP 报文不仅传送用户数据, 还包括发送方和接收方的协议端口号, 以使接收方的 UDP 软件能将报文送到正确的接收进程, 并回送应答报文给对应的发送进程。

UDP 使用底层的网际协议 (IP) 来传送报文, 同 IP 一样, 提供不可靠的无连接数据报传输服务。它不具备报文到达确认、排序以及流量控制等功能, 因此报文可能会丢失、重复以及乱序等。而可靠性的问题将由使用 UDP 的应用程序自己来解决。

每个 UDP 报文称为一个用户数据报, 分 UDP 报头和 UDP 数据区两部分。UDP 位于 IP 层之上。应用程序访问 UDP 层, 然后使用 IP 层传送数据报。将 UDP 层放到 IP 层之上, 表示一个 UDP 报文在互联网中传输时要封装到 IP 数据报中。最后, 网络接口层将数据报封装到一个帧中再进行物理传输通道上的传输。

IP 层的报头指明了源主机和目的主机的地址,

而 UDP 层的报头指明了主机上的源端口和目的端口。

UDP 也提供复用和分解的功能。它接收多个应用程序送来的数据报,把它们送给 IP 层去传输,同时它接收 IP 层送来的 UDP 数据报,把它们送给对应的应用程序。

从概念上讲,所有的 UDP 软件与应用程序之间的复用和分解都要通过端口机制来实现。实际上每个应用程序在发送数据报之前必须与操作系统进行协商以获得协议端口和相应的端口号。凡是利用指定的端口发送数据报的应用程序都要把端口号放入 UDP 报文中的源端口字段中。

UDP 端口号的指定有两种方式:一种是由某些管理机构指定的,称为著名端口,供用户使用;另一种是动态绑定方式,由应用程序指定端口。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999

2. Douglas E Comer. Internetworking with TCP/IP, Volume I, 3rd ed. Prentice Hall Inc., 1995

(胡道元)

youxianyuan fangfa

有限元方法 (finite element method) 求解二维或三维连续体内偏微分方程(特别是椭圆型方程)边值问题的一种数值计算方法。该法的基础为变分原理和剖分插值,前者是把需求解的偏微分方程化为等价的泛函极小问题,后者则是把求解区域的连续体剖分成有限多个小的单元体,并用简单的分片插值函数来代替所求的未知函数,这两者相结合,即把一个在连续体区域上求解偏微分方程的问题化成为在区域内部和边界的一组离散点集上求解代数方程组问题,这种离散化方法称之为有限元方法。它与传统的有限差分离散方法比较有以下显著的优点:①适合于任意复杂的求解区域以及区域内不同介质的处理。②边界条件处理方便自然。③离散方法统一,特别适合于大型电子计算机编制程序,现已研制出许多通用的有限元分析软件包提供用户使用。

变分原理 许多椭圆型方程边值问题,都与一个适当的“泛函极小原理”(变分原理)相等价,这样就把一个求解偏微分方程问题化为一个求解泛函极小问题。如二维弹性力学偏微分方程为

$$\left. \begin{aligned} \frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + f_x &= 0 \\ \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_y}{\partial y} + f_y &= 0 \end{aligned} \right\} (x, y) \in \Omega_1 \cup \Omega_2 \quad (1)$$

外力边界条件为

$$\left. \begin{aligned} \sigma_x \cos(n, x) + \tau_{xy} \cos(n, y) &= \varphi_x \\ \tau_{xy} \cos(n, x) + \sigma_y \cos(n, y) &= \varphi_y \end{aligned} \right\} (x, y) \in \Gamma_1 \quad (2)$$

位移边界条件为

$$\left. \begin{aligned} u &= \bar{u} \\ v &= \bar{v} \end{aligned} \right\} (x, y) \in \Gamma_2 \quad (3)$$

弹性系数间断边界条件为

$$\left. \begin{aligned} &(\sigma_x \cos(n, x) + \tau_{xy} \cos(n, y))^+ \\ &= (\sigma_x \cos(n, x) + \tau_{xy} \cos(n, y))^- \\ &(\tau_{xy} \cos(n, x) + \sigma_y \cos(n, y))^+ \\ &= (\tau_{xy} \cos(n, x) + \sigma_y \cos(n, y))^- \end{aligned} \right\} (x, y) \in \Gamma_0 \quad (4)$$

上述偏微分方程(1)及边界条件(2)至(4)等价于变分原理(最小势能原理)

$$\begin{aligned} J(u, v) &= \frac{1}{2} \iint_{\Omega \cup \Omega_2} \sigma^T \varepsilon dx dy - \iint_{\Omega \cup \Omega_2} f^T \delta dx dy \\ &- \int_{\Gamma_2} \varphi^T \delta ds = \min \end{aligned} \quad (5)$$

这里

$$\sigma = \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix} = D \begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{Bmatrix}, \quad \varepsilon = \begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{Bmatrix} = \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \end{Bmatrix} \quad (6)$$

分别为应力应变分量, $\delta = \begin{Bmatrix} u \\ v \end{Bmatrix}$ 为位移分量, D 是根

据虎克定律得到的弹性矩阵 $f = \begin{Bmatrix} f_x \\ f_y \end{Bmatrix}$, $\varphi = \begin{Bmatrix} \varphi_x \\ \varphi_y \end{Bmatrix}$ 分别为给定的体积力和 Γ_1 上的边界力, $(\cos(n, x), \cos(n, y))$ 为边界曲线法线的方向余弦, 弹性系数在 Ω_1, Ω_2 内可分别有不同的值, 对式(5)中的 u, v 取极小, 则满足式(5)的 u, v 除了在区域 $\Omega_1 \cup \Omega_2$ 内满足平衡方程(1)外, 在边界 Γ_1, Γ_0 自动满足边界条件(2)~(4)。

剖分与插值 为适于电子计算机数值求解, 还必须对上述式(5)进行离散化, 为此, 需对求解区域 Ω 进行单元剖分, 在每个单元内用一个较简单的

插值函数代替未知函数 u, v , 在二维区域内最基本最简单的单元是三角形单元 (见图 1)。在曲线边界上则裁弯取直, 用折线代替曲线, 当然还可以取四边形单元或曲边的三角形和四边形单元等, 这些单元的顶点可称为网格结点。插值函数依据不同的单元类型有不同的取法, 对于简单的三角形单元, 使用线性插值函数, 即在每个三角形单元 Δ_k 内, 用 $U(x, y) = a_k x + b_k y + c_k$ 来近似代替 $u(x, y)$, 其中系数 (a_k, b_k, c_k) 由条件 $U_i = U(x_i, y_i) = U_i (i = 1, 2, 3)$ 来确定。这样在区域 Ω 内部, 未知函数 $u(x, y)$ 用一个分片线性函数 $U(x, y)$ 代替, 对于未知函数 $v(x, y)$ 按同样方法处理。

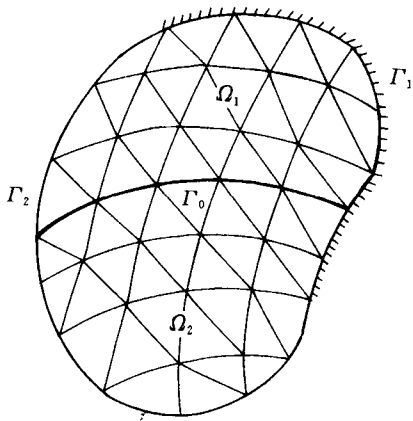


图 1 三角形单元

有限元的离散化 上述插值函数 $U(x, y)$, $V(x, y)$ 取定以后, 将其代入式 (6), 则有

$$\epsilon = \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{Bmatrix} = B\Delta, \quad \sigma = D\epsilon = DB\Delta \quad (7)$$

这里 B 为应变-位移矩阵, Δ 为所有网格结点上位移值 $\begin{Bmatrix} U_i \\ V_i \end{Bmatrix}$ 组成的向量, 把式 (7) 代入式 (5) 有

$$\begin{aligned} J(u, v) &\approx J(U, V) = \frac{1}{2} \iint_{\Omega} \epsilon^T D \epsilon dx dy - \Delta^T F \\ &= \frac{1}{2} \Delta^T K \Delta - \Delta^T F \end{aligned} \quad (8)$$

这里 $K = \iint_{\Omega} B^T D B dx dy = \sum_e \iint_{\Omega} B^T D B dx dy = \sum_e K_e$, K_e 称为单元刚度矩阵, K 为整体刚度矩阵, 它是由各单元刚度矩阵迭加而得, F 是由式 (5) 中第二、三项离散后得到的外力向量, 在式 (8) 中对 Δ 中所有分量求偏导数, 并令其等于 0, 得到下面的线性代数

方程组

$$K\Delta = F \quad (9)$$

从上面过程可以看到, 在变分原理及剖分插值函数取定以后, 线性方程组 (9) 即完全确定, K 和 F 的具体计算由电子计算机以统一的方式完成, 可以很方便地进行计算机程序编制。

有限元数值解法的应用 从上面得到的线性代数方程组 (9), 其系数矩阵 K 有许多好的性质, 如对称、正定和高度稀疏性, 这就为数值求解提供了很好的条件, 求解这种线性方程组的方法, 归纳起来有两类: 一类是直接解法, 如线性方程组的高斯消去法, 平方根法。另一类是迭代解法, 如逐次超松弛法, 切比雪夫半迭代以及带预条件 z 的共轭斜量法 (PCG 法) 均是目前常用的有效的数值解法。当求得结点位移以后, 将其代入应力-应变-位移关系式 (7), 可得到工程设计中最感兴趣的各个单元应力和结点应力值。

由于有限元方法能方便求解各种比较复杂的问题, 加之计算机的飞速发展, 大容量的存储及快速运算速度, 促进了有限元方法在几乎所有工程部门 (如航空航天、水工建筑、机械电子、石油化工以及国防工业部门) 的应用, 而针对各个工程部门研制出了各种大型通用和专用有限元软件包 (如美国的 ADINA, ANSYS, NASTRAN, 德国的 ASKA 等), 与现代先进的计算机辅助设计技术 (CAD) 相结合, 大大地缩短了工程设计周期, 极大地推动了生产力发展与科学技术进步。

有限元方法最早由 R. Courant 于 1943 年在一篇论文中提出, 由于其运算量大, 没有受到重视和得到应用发展。直到 50 年代中期, 随着电子计算机出现, 在西方首先由工程师们应用于航空结构分析, 以后随着计算机技术的发展而逐步应用于其他工程领域, 60 年代后期, 数学家们也相继参加到有限元方法研究中来, 逐步建立了有限元方法严格的数学理论。在这里应当提到的是我国科技工作者对有限元方法应用与发展也作出了重要贡献。60 年代初, 我国冯康教授及其领导下的科研组在解决大型水坝计算中, 独立于西方发展了有限元方法, 并且先于西方国家建立了严格的有限元的数学理论基础, 以后我国许多科技工作者又在许多具体方向上, 作出了很多有价值的贡献。

参考文献

1. Zienkiewicz O Z. The Finite Element Method, 3rd ed. London: McGraw - Hill, 1977

2. 冯康. 数值计算方法. 北京: 国防工业出版社, 1978 (王萃贤)

youxian zidongji

有限自动机 (finite automaton, FA) 有限离散数字系统的抽象数学模型。加法器、奇偶校验器、模 3 计数器、二单位时间延迟器、自动电话交换机、升降机、计算机操作系统进程的活动过程、计算机、时序电路、开关网络等都是有限离散数字系统的实例。主要研究系统的综合与分析, 即给出具体的功能要求, 综合建成能实现这些功能的有限自动机并设计出满足要求的逻辑网络, 给出有限自动机或其实现网络, 分析有限自动机的功能描述, 时间复杂度, 故障传播与检测。

有限自动机的类型与等价 确定型有限自动机 (DFA) 的数学定义为五元组 $M = (Q, \Sigma, \delta, q_0, F)$, 其中 Q 是状态的有限集, Σ 是输入字母的有限集, $q_0 \in Q$ 称为初 (始) 状态, $F \subseteq Q$ 为终 (止) 状态集, 转移函数 $\delta: Q \times \Sigma \rightarrow Q$ 是单值函数, 即对任一输入来说, 在给定状态下有惟一的下一步的状态。实际问题中常常要求处理字母的有限序列 (包括空序列 ε , 或称空字), 故 δ 的定义范围往往扩大到集合 Σ^* 上。若 $\delta: Q \times \Sigma^* \rightarrow Q$ 是 (全) 函数, 称 M 为完全的确定型有限自动机; 若 δ 是部分 (偏) 函数 (对有些状态无定义) 称为不完全的 (偏) 确定型有限自动机。若 $\delta: Q \times \Sigma^* \rightarrow 2^Q$ (2^Q 表示 Q 的所有子集的集合即幂集), 即 δ 为多值函数, 称 M 为非确定有限自动机 (NFA), 也可分为完全的和偏的两种。若 $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$, 称 M 为带 ε -转移的 NFA。若 $\delta: Q \times \Sigma \rightarrow Q \times \{L, R\}$ 称 M 为双向有限自动机, 其中 L, R 分别表示扫描输入字母时读头左, 右移一个位置 (2 DFA)。已经证明 DFA, NFA, 带 ε -转移的 NFA, 2DFA 等价, 即功能相同。实际问题 (如寄存器、计数器、信号转换器等) 常常需要考虑输出的结果和状态, 故研究了时序机械装置——两类带输出的有限自动机。①六元组 $M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ 称为摩尔机 (状态赋值机), 其中 Q, Σ, δ, q_0 与 DFA 中相同, Δ 为输出字母的有限集, $\lambda: Q \rightarrow \Delta$ 是单值函数, 即输出只与到达的状态有关, 与从哪个状态转移而来无关。输入 ε 时, 输出为 $\lambda(q_0)$, 输入 $a_1 a_2 \cdots a_n$ 时输出为 $\lambda(q_0) \lambda(q_1) \cdots \lambda(q_n)$, 其中 $\delta(q_{i-1}, a_i) = q_i, 1 \leq i \leq n$ 。DFA 可以看作 $\Delta = \{\text{接收}, \text{拒绝}\}$ 时的摩尔机。②六元组 $M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$, 称为米林机 (转换赋值机), 其中 $\lambda: Q \times \Sigma \rightarrow \Delta$, 其余与摩尔机中相同。

输入 ε 时输出也为 ε , 输入 $a_1 a_2 \cdots a_n$ 时, 输出为 $\lambda(q_0, a_1) \lambda(q_1, a_2) \cdots \lambda(q_{n-1}, a_n)$, 其中 $q_i = \delta(q_{i-1}, a_i), 1 \leq i \leq n$ 。已知摩尔机与米林机等价。有限自动机有两种表示法, 一是状态表或转移矩阵, 一是有向图 (称为状态图用 \odot 表示终态)。例如奇偶校验器可设计为一台米林机 $M_1 = (\{A, B\}, \{0, 1\}, \{0, 1\}, \delta, \lambda, A)$, 输入串有奇数个 1 时输出 1, 输入串有偶数个 1 时 (此时状态为 A) 输出 0。 M_1 的状态表和状态图如图 1 所示。

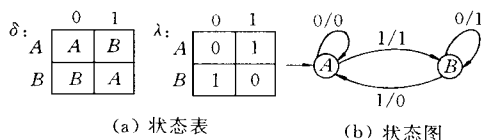


图 1 M_1 的状态表和状态图

米林机状态图中弧上带的符号为输入/输出。从上述米林机可设计一台摩尔机 $M_2 = (\{A, 0\}, \{B, 1\}, \{0, 1\}, \{0, 1\}, \delta, \lambda, (A, 0))$ 。 M_2 的状态表和状态图如图 2 所示。

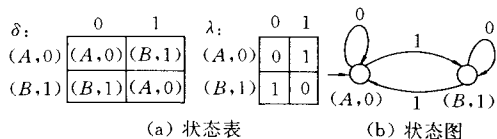


图 2 M_2 的状态表和状态图

状态化简 实际问题迫切需要设计出功能相同而状态最少的系统, 这就是有限自动机的极小问题。对完全确定有限自动机的状态集引入关系“ \equiv ”, 对每一输入串 x , 若 $\delta(p, x) \in F$ 当且仅当 $\delta(q, x) \in F$, 则称状态 $p \equiv q$, “ \equiv ”是一个等价关系, 从而可将 Q 划分为若干个等价类, 凡在同一个等价类的状态具有相同的功能, 不在同一等价类的状态功能不同, 于是可简化地视一个等价类为一个状态, 从而完全的确定型有限自动机在同构的意义下有惟一的与它功能相同而状态数最少的有限自动机存在, 从而给出了极小化的有效算法。

时序网络与有限自动机的模拟 组成时序网络的基本逻辑元件有组合元件与记忆元件两类。组合元件在时刻 t 的输入完全决定了时刻 t 的输出, 如与、或、非、与非、或非等阈电路元件。记忆元件在时刻 t 的输出与时刻 t 的输入与状态有关, 如各种触发器和延迟器。去掉时序网络中连接记忆元件的网络输入线后不再含有回路, 称之为合式网络, 无记忆元

件的合式网络称为组合网络。用布尔代数或卡诺图可简化描述输入输出方程组,设计出符合要求、结构简单的二值元件组合网络。合式网络需要用极小化有限自动机。具有同样输入集和输出集的两个有限自动机 M_1 和 M_2 的状态 q_1 和 q_2 ,若对所有输入序列都有相同的输出,称 q_1 与 q_2 等价。如果 $M_1(M_2)$ 的每个状态总有 $M_2(M_1)$ 的一个状态与之等价,称 M_1 与 M_2 等价,即 M_1 与 M_2 功能相同。现作进一步推广,若有穷自动机 $M_1 = (Q_1, \Sigma_1, \Delta_1, \delta_1, \lambda_1, q_0)$ 与 $M_2 = (Q_2, \Sigma_2, \Delta_2, \delta_2, \lambda_2, q_0')$ 之间存在映射 $h_1: Q_2 \rightarrow Q_1$, 映射 $h_2: \Sigma_2 \rightarrow \Sigma_1$, $h_3: \Delta_1 \rightarrow \Delta_2$, 且对 $a_2 \in \Sigma_2, q_2 \in Q_2$ 时有

$$\begin{aligned} h_1^{-1}(\delta_1(h_1(q_2), h_2(a_2))) &= \delta_2(q_2, a_2) \\ h_3(\lambda_1(h_1(q_2), h_2(a_2))) &= \lambda_2(q_2, a_2) \end{aligned} \quad (1)$$

称 M_1 是 M_2 的一个模拟。映射 h_2 是一个编码器,它将 M_2 的输入信息变成 M_1 的输入信息,然后再输入 M_1 ,应用转移函数 δ_1 处理后输出,经译码器 h_3 译成 M_2 的输出信息。若满足式(1), M_1 就是 M_2 的一个模拟。 h_2, h_3 不同,模拟就有所不同。反映在有限自动机模拟时序电路也就不同,即模拟时对每一状态指定一组代表基本元件状态组合的编码,将编码分配给各状态。不同的分配方案产生不同的逻辑网络,各有其复杂程度。选择好的状态分配方案使逻辑网络的构造尽可能简单,是模拟的主要研究课题之一。加上延迟器的异步网络,考虑基本元件状态组合的编码分配时,还需要注意状态转换的稳定性,保证无矛盾、无竞争。

有限接收器 不考虑输出的有限自动机 $M = (Q, \Sigma, \delta, I, F)$, 其中 I 为初态集,字 $x \in \Sigma^*$ 合于 $\delta(q_0, x) \in F, q_0 \in I$, 称 x 为 M 所接收或 M 能识别 x 。 $L(M) = \{x \in \Sigma^* | \delta(q_0, x) \in F, q_0 \in I\}$ 称为 M 所接收(识别)的语言,是正规语言(它由某一正规文法产生,与正规集或正规表达式等价)。任一正规语言可以由一个带 ε -转移的不确定有限自动机所接收。有限自动机和有限接收器有三点不同:①前者只有一个初态;后者为非空初态集。②前者主要考虑对输入的响应;后者考虑的是输入后的状态是否属于终态集,以便判断该输入是否被接收。③前者的转移函数往往是单值的;后者是多值的且可能是偏函数。

在与形式语言乔姆斯基分层对应的自动机分层中,有限自动机类是下推自动机类的真子集。有限自动机经布尔运算(并、交、补),替换,同态,逆同

态,商, MAX, MIN, CYCLE, INIT, reversal, $1/2$, SQRT, LOG 等运算后仍为有限自动机。

神经网络 历史上最早用有限自动机模型化的系统。每一个神经元由细胞体、神经纤维和突触组成,细胞体经神经纤维发出若干突触。有限个神经元连接在一起构成神经网络。连接时,每个神经元的任一突触只接触一个细胞体。每一突触只有兴奋(用圈表示)和抑制(用点表示)两种状态。若输入 1, 兴奋突触个数超过抑制突触的个数,至少为该细胞体(用三角形表示)的阈值(置于细胞体中),神经元就输出 1。假定有足够的时间改变输入值使信号传播,网络达到稳定的配置,则等价于图 3 所示神经网络行为的有限自动机(y_1, y_2, y_3 的初值设为 0)如图 4 所示。

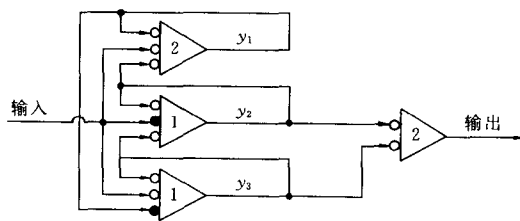
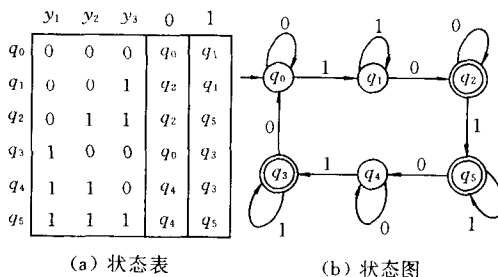


图 3 神经网络



(a) 状态表

(b) 状态图

图 4 与图 3 相应的有限自动机

有限自动机的应用 计算机程序语言编译中,话前缀的集合是正规语言,可以被生成该语言之文法的项目作为状态并另加一初态的非确定的有限自动机所接收,然后找出等价的极小确定有限自动机,从而节约词法分析的时间和空间。又在编译程序里可完成扫描器的任务,识别变元名字、运算符、常数、标识符等。例如长度限制为 6 的 FORTRAN 语言标识符和 \$ 以及大写英文字母可表示为正则表达式

$$(\$ + A + B + \cdots + Z)(\varepsilon + \$ + A + B + \cdots + Z + 0 + 1 + \cdots + 9)^5$$

转移函数用二维数组编码表表示,终态表明已找到

特殊的符号。

文本编辑程序也与词法分析的情形类似,将字符串与一正规表达式相匹配,求接收它的带 ε -转移的非确定有限自动机,一旦列举该自动机所有状态的列构造好,就使之在输入的前缀上进入,前面的列不再需要便舍弃掉以节约空间。可应用于图书资料索引查找,结构型数据翻译等。

是否能够由输出决定相应的输入呢?这个问题导致研究有限自动机的可逆性。现已得到可逆的充分必要条件是有限自动机的状态图 G 没有回路, G 的层次为 ρ ,则自动机延迟 $\rho+1$ 步可逆,且对任何 $\tau \leq \rho$,不能延迟 τ 步可逆。

当有限自动机 M 的输入、输出、状态集分别为有限域上 l, m, n 维向量空间时,称 M 为线性有限自动机。当有限自动机 $M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ 的转移函数 δ 和输出函数 λ 不依赖于输入时,称为自治有限自动机。对它们也证明了存在其极小化。对编码、移位寄存器、通信的可靠性、保密性的研究有其作用。近年来也研究了状态识别试验,即在不知道有限自动机 M 的内部状态时,通过馈入序列来观察分析 M 的结构与功能,现已发展成暗箱理论。

也有人去掉输入字长有限这一条件,研究 ω -有限自动机,平行地得到了一些结果。

参考文献

Hopcroft J E and Ullman J D. Introduction to Automata Theory, Languages, and Computation. Reading, Mass.: Addison-Wesley, 1979 (张一立)

youxiangtu

有向图 (directed graph) 每条边均为有向边的图(参见图论)。设有向图 $G = \langle V, E, \psi \rangle$,对任意顶点 $v \in V$,称以 v 为终点的边的数目为 v 的入度,记为 $d_c^-(v)$ 。称以 v 为起点的边的数目为 v 的出度,记为 $d_c^+(v)$ 。称 $d_c^+(v) + d_c^-(v)$ 为 v 的度,记为 $d_c(v)$ 。如果 G 中任意两顶点都相互可达,则称 G 是强连通的。 G 的极大强连通子图称为 G 的强分支。如果对于 G 的任意两顶点,必有一个顶点可达另一个顶点,则称 G 是单向连通的。 G 的极大单向连通子图称为 G 的单向分支。如果 G 的底图是连通的,则称 G 是弱连通的。 G 的极大弱连通子图称为 G 的弱分支。设 $n \in I_+$ (正整数集合),每个顶点的出度和入度均为 $n-1$ 的 n 阶简单有向图称为完全有向图。设 $V = \{v_1, v_2, \dots, v_n\}$ 且 $E = \{e_1, e_2, \dots, e_m\}$, G

的邻接矩阵 $X(G)$ 是一个 $n \times n$ 矩阵 (x_{ij}) ,其中 x_{ij} 为以 v_i 为起点和以 v_j 为终点的边的数目。 G 的路径矩阵(或称可达性矩阵) $P(G)$ 是一个 $n \times n$ 矩阵 (p_{ij}) ,其中,

$$p_{ij} = \begin{cases} 1, & \text{若从 } v_i \text{ 可达 } v_j \\ 0, & \text{否则} \end{cases}$$

若 G 无自圈,则 G 的关联矩阵 $A(G)$ 是一个 $n \times m$ 矩阵 (a_{ij}) ,其中

$$a_{ij} = \begin{cases} 1, & \text{若 } v_i \text{ 是 } e_j \text{ 的起点} \\ -1, & \text{若 } v_i \text{ 是 } e_j \text{ 的终点} \\ 0, & \text{否则} \end{cases}$$

当然, G 还有其他的矩阵表示。矩阵表示很适合用计算机表示有向图和对有向图进行操作。(张强)

yufa

语法 (syntax) 构建语言的结构正确成分所需遵循的规则集合。**程序设计语言**的语法定义阐明了何种字符串构成该语言的一个合法的程序。早期语言的语法是用自然语言描述的,如早期的**FORTRAN**语言。从**ALGOL 60**语言开始,人们都用形式方法描述语法。描述语法的常用形式描述工具是**BNF**,扩充的**BNF**和**语法图**。(程虎)

yufa fenxi

语法分析 (syntax analysis, parsing) 按语言的语法规则分析和处理由词法分析程序产生的词牌流并转换成语法分析树的过程。

程序设计语言的语法一般是上下文无关文法,即chomsky 2型文法。通常采用**BNF**或**语法图**来描述。

语法分析是编译过程的一个阶段,其输入为词牌流,输出为语法分析树的某种表示及有关信息。有些编译程序,在语法分析过程中除对源程序进行语法检查外,还进行静态语义检查。当检查中发现不符合规则的情况时,系统应指明错误的性质和可能的错误位置,以供用户改错时参考。

实现语法分析的算法有多种,如算符优先法、递归下降法和LR法等。其中一类为自顶向下分析方法,即从树根(程序)向树叶(即终结符)方向进行分析。递归下降法是一种典型的自顶向下分析方法。另一类为自底向上分析方法,即从树叶向树根方向进行归约。算符优先法是一种典型的用来处理表达式的自底向上分析方法。LR分析法可以认为是自

左至右、自底向上的移进-归约分析的高度概括和集中,它比算符优先法或其他“移进-归约”技术更加广泛,而且识别效率并不比它们差。

参考文献

1. Aho A V, Sethi R, Ullman J D. Compilers Principles, Techniques and Tools. Addison-Wesley, 1986

2. 陈火旺,钱家骅,孙永强. 程序设计语言编译原理(第二版). 北京:国防工业出版社,1984

(钱树人)

yufatu

语法图 (syntax diagram) 语法的图形描述。每个图描述一种非终极符号的定义。图中用圆圈表示终极符号,用方框表示非终极符号,用有向弧表示走向,图上一条通路就表示该语法结构的一种正确定义方式。

例如,某语言的标识符可描述为图 1(a),表示由字母开头的字母数字串组成,字母和数字最多总共可有 6 个。字母可描述为图 1(b),数字可描述为图 1(c)。

(程虎)

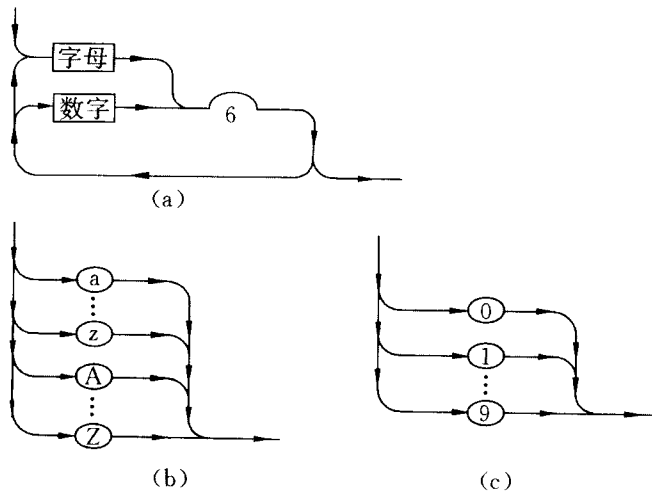


图 1 语法图

yujing fenxi

语境分析 (context analysis) 在自然语言理解中根据语言单位出现的环境来排除歧义的方法。在语言学界,不同学派对“语境”这个概念有不同的术语和定义。“context”这个词在英文中的原意是指文章或言谈中某一句话的上句或下句,一段话的上段或下段,所以又译作“上下文”。但是后来人们把这个词的意义扩展了,既指任何一级语言单位出现的语言环境,也指文章或言谈的社会环境,即是何时何地、谁对谁所说的话。不过各学派都公认,任何一级语言单位(包括字、词、短语、句子、句群和篇章等)一旦进入使用,它就不但受到周围其他语言单位和语言使用环境的约束,而且也将对其他语言单位产生一定的限制。所以,一般来说一个语言单位只有放到更大的语言环境中去,才能被正确地理解。

自然语言理解和机器翻译中的语境分析包含以下两方面的内容:

(1) 利用上句或下句的信息来排除一个句子的句法结构歧义。在英语中,介词短语究竟是修饰前面的动词还是名词,常常需要靠语境分析来确定。例如,下面的句子:

“I saw a man with a telescope.”

有两种解释:

“我看见一个带望远镜的男人。”

“我用望远镜看见一个男人。”

而正确的解释只能依赖于上句或下句提供的进一步的信息。汉语中也常有类似的结构歧义,如:

“学校里来了三个学生的家长。”

句中“三个”是指学生还是家长?也需要通过语境分析来辨识。

(2) 在话语理解(或分析)中,需要利用上下文的知识来解决前指、后指、省略、连贯等问题。

这些仅仅依靠在一个句子的范围内进行句法、语义分析无法消解的歧义问题,只能通过语境分析,

即超越句子范围的分析,来加以解决。所以语境分析在自然语言理解中有其重要意义。

参考文献

西楨光正. 语境研究论文集. 北京: 北京语言学院出版社, 1992 (黄昌宁)

yuju

语句 (statement) 高级语言中用于表达处理动作的基本单位,用以描述程序中的运算步骤、控制构造及数据传输。

从语句的构造方式来看,语句可分为简单语句与构造语句。简单语句包括空语句,赋值语句,过程调用语句(又称过程语句),转向语句,出口语句,返回语句等;构造语句包括复合语句,重复语句,条件语句等。

赋值语句将表达式的值作为左部变量的当前值。其形式是

变量名: = 表达式 (PASCAL 语言形式)

变量名 = 表达式 (FORTRAN 语言形式)

一般说来,要求赋值变量的类型与右部表达式值的类型相同或赋值相容。

过程语句表示对过程名所标记的过程的一次调用,其形式是

过程名(实在参数表)

实在参数表可以无参数,也可以有一个或多个参数。

过程语句的执行首先检查参数表是否一致,即其实在参数表中的实参与过程说明中的形参在个数、顺序、类型上是否保持一致。若一致,则传递参数,然后去激活相应的过程体(参见**函数与过程**)。

空语句表示空运算;转向语句为用以改变程序执行次序的语句。

构造语句是由简单语句按一定规则构造起来的语句。

复合语句是将若干个语句用 begin 和 end 括起来的一个语句。程序执行时按书写的顺序执行复合语句内的各个成分语句。

条件语句一般又可分为如果语句与情况语句两种。如果语句主要用于处理有两个分支的选择结构,而情况语句主要用于多分支选择结构。

如果语句有 if-then 和 if-then-else 两种形式。

如果语句规定:在一定条件下执行某一种成分语句,否则执行另一种成分语句;或者不执行任何成分语句。

情况语句的一般形式为:

case 表达式 of

情况常量表 1: 语句 1;

情况常量表 2: 语句 2;

...

情况常量表 n: 语句 n

end

情况语句就是根据情况表达式的取值,选择其值与情况常量表的某一情况常量相同的成分语句执行,该成分语句执行后,便去执行该情况语句后的语句。

重复语句规定一些语句被重复地执行。被重复的语句称为重复体。重复语句包括 while 语句, Repeat 语句和 for 语句。

while 语句的一般形式为:

while 条件 do 语句

其中,条件为一布尔表达式,do 后面的语句是 while 语句的重复体,它可以是简单语句,也可以是复合语句。

Repeat 语句的一般形式为:

Repeat

语句 1;

语句 2; 重复体

...

语句 n

until 条件 (布尔表达式)

重复执行重复体内的语句,直到 until 后的布尔表达式的条件成立为止。

for 语句有 for-to-do 和 for-down to-do 两种形式。

for 语句是预先给出重复次数的一种重复语句,其重复次数由指定的初值和终值决定,并由循环控制变量来控制。循环控制变量在指定的值域[初值,终值]内顺序取值(分递增与递减两种方式),每取一次值就执行一次重复体,直至该值域内的值全被访问,重复结束。

从语句的执行方式来看,又可分为顺序语句与并发语句两类,与顺序执行的语句不同,并发语句用来描述若干个并发的计算活动,这些活动的总体效果与活动开始的次序或活动执行的次序无关。引入并发语句的第一个语言为 P. B. Hansen 开发的并发 Pascal,它是 **Pascal 语言** 的一个变种。语言中的复合语句,子程序及程序按顺序语言的分程序结构方式组织起来,但处理部件要等确立并发性的其他程序触发之后,并让它们真正启动才可以执行。以下

是一个由三个过程组成的复合体的例子,它们中一个或全部均能并行地执行。

```

type P1 = Process (相关部件表)
Var      P1 的说明
begin    P1 的活动      P1 定义
end.
type P2 = Process (相关部件表)
Var      P2 的说明
begin    P2 的活动      P2 定义
end.
type P3 = Process (相关部件表)
Var      P3 的说明
begin    P3 的活动
end.
Var      Pname1:P1;      Pname2:P2;
          Pname3:P3;      激活并发进程
begin init Pname1,Pname2, Pname1,Pname2,
          Pname3;      Pname3.
end.

```

这里 Pname1, Pname2 及 Pname3 分别为 P1 类型, P2 类型及 P3 类型的实例。

早期有些高级语言(如 FORTRAN IV)中,把语言中不是用来表达算法,而是用来描述数据对象的定义的说明性成分也看作为语句,把这类语句称为不可执行语句,并把以前的表达动作行为的语句称为可执行语句。现代高级语言中,一般将这类说明性的不可执行语句列入语言的说明部分(参见说明)。

参考文献

Ralston A, Reilly E D. Encyclopedia of Computer Science. third edition. New York: Van Nostrand Reinhold, 1992
(陈涵生)

yuliaoiku

语料库 (corpus) 用于语言研究和语言工程的语言材料的汇集。它通常是以电子文件或数据库形式存放在计算机存储介质上的,大规模的组织良好的语言材料。

教育学家陈鹤琴于 20 世纪 20 年代收集 6 类计 55 万汉字的文本,据此整理了《语文体应用字汇》,首开汉语语料库建设与应用的先河。早期语言学家常以卡片形式手工采集语料,语料库的规模小,影响也小。当代计算机、网络及其应用技术普及之后,采集语料容易了,面向不同应用的各类语料库便迅速

发展起来。我国语料库建设及其相关研究也受到国际潮流的影响,其中英语语料库的影响最大。

语料库分类 多数语料库收集书面语言,也有口语语料库(录音再转成文字材料);有些在命名上冠以“平衡”,以表示其语料的选择考虑了不同内容、不同语体的适当比例,也有些语料库只聚焦于某个领域或某个来源;区别于单语语料库,双语或多语语料库不仅收集两种以上语言的语料,而且在不同语言之间建立起对应关系;对所包含的语料有没有进行加工成为区分语料库的最重要的标志,原始语料库未对文本进行任何加工,尽管按照收录的单位(如篇章)可能附加一些用于管理的属性信息(如年代、领域、作者等);加工语料库则对语料本身标注了语音、句法、语义等信息。

语料库加工 汉语语料库最受重视的加工项目包括词语切分、词性标注、短语结构或句法树标注、词语的义项标注等。2002 年教育部发布《第一批异形词整理表》。整理异形词所遵循的通用性原则立足于词频统计。“毕恭毕敬”和“必恭必敬”是一组异形词,在 1.5 亿字《人民日报》语料中统计,“毕恭毕敬”出现了 24 次,而“必恭必敬”为 0 次,于是选择“毕恭毕敬”作为推荐词形。词频统计对任何语言都是一项最基本的语言工程。但中文文本书写方式(词与词之间不留空格)给词频统计带来了困难。“过分”和“过份”也是一组异形词。只在文本中简单地统计汉字串“过分”的次数,就会出现误差。像“有人做过分析”、“他们有过分歧”、“已经检查过分会场了”等等汉字串中的“过分”都不是一个词。汉字串“过分”的频度并不等于“过分”这个词的频度。汉语词频统计的基础是实现大规模文本的词语切分。汉字相同的词形可能是词性、读音或意义并不相同的词。以“把”为例,有介词、量词、动词 3 种词性,意义和用法都不同。在切分的基础上标注词性,便可以得到附加词性的词频。进一步标注拼音,则可提供多音词在文本中的正确的读音知识。

对原文“咱们中国这么大的一个多民族的国家如果不团结,就不可能发展经济,人民生活水平也就不可能得到改善和提高。”进行切分、标注、注音,可以得到加工语料库的如下样例:

咱们/r 中国/ns 这么/r 大{da4}/a 的
{de5}/u 一个/mq 多/a 民族/n 的{de5}/u
国家/n 如果/c 不/d 团结/a ,/w 就/d 不/
d 可能/v 发展/v 经济/n ,/w 人民/n 生
活/n 水平/n 也/d 就/d 不/d 可能/v 得

到/v 改善/vn 和{he2}/c 提高/vn 。/w

经加工后,词与词间有了空格,斜杠右边的字母是词性标记:r——代词,ns——地名,a——形容词,u——助词,mq——数量词,n——名词,c——连词,d——副词,w——标点符号,v——动词,vn——名动词。斜杠左边花括号内的字母数字串是多音词的汉语拼音(分别用1,2,3,4,5代表4声和轻声)。

大规模的经过如此加工的语料,无论在信息处理领域,还是在汉语本体研究以及汉语教学领域都有重要的应用价值。

语料库语言学 以语料库为基础进行语言研究的语言学的分支学科。由于它为自然语言处理提供基于统计的理论模型和算法,并取得显著成就而日益受到重视,常被认为是计算语言学的分支。不过,语料库语言学家更重视语料库分析对认识语言可能带来的独特贡献,强调语言理解涉及人类的意识活动,不能变换为在计算机上实现的程序。这种认识把语料库语言学和计算语言学区别开来。

最常用的基于统计的语言模型是 n 元语法。 n 元语法的目标是计算由词串 $w_1 w_2 \cdots w_i \cdots w_n$ 组成的语句 s 出现的概率 $P(s) = P(w_1 w_2 \cdots w_i \cdots w_n)$ 。

理论上,统计词串 $w_1 w_2 \cdots w_i \cdots w_n$ 在大规模文本(设词语总量为 N)中出现的次数 f ,其频率 f/N 可以看作是 $P(s)$ 的近似值。不过,“数据稀疏”是 n 元语法的致命弱点:设不同的词只有 5 000 个且 $n=3$, 3 个词的不同组合有 1 250 亿。随机排列的 3 个词 $w'_1 w'_2 w'_3$ 很可能不出现在千万字量级的语料库中,得到 $P(w'_1 w'_2 w'_3) = 0$,而实际上 $P(w_1 w_2 w_3) \neq 0$ 。为了克服“数据稀疏”,可按照某种准则将词的集合划分为类的集合,以类元素排列替代词本身的排列。词的语法分类(如名词 n 、动词 v 、形容词 a 、副词 d 等)是最常用的划类方法。词类集合中的元素(或者说,词性)排列的序列反映了一定的句法结构,基于语法词类的 n 元语法的应用比较普遍。

设标注语料库词性标记集合中的元素为 $c_j (j=1, 2, \cdots, m)$,面向类的 n 元语法模型要求计算 $P(s) = P(c_1 c_2 \cdots c_i \cdots c_n)$,即词性标记序列 $c_1 c_2 \cdots c_i \cdots c_n$ 的概率。

利用概率论的知识,可以化简 $P(c_1 c_2 \cdots c_i \cdots c_n)$ 的计算公式。

$$P(c_1 c_2 \cdots c_i \cdots c_n) = P(c_1) P(c_2 | c_1) P(c_3 | c_1 c_2) \cdots P(c_n | c_1 c_2 \cdots c_{n-2} \cdots c_{n-1})$$

式中: $P(c_i)$ 是 c_i 的概率, $P(c_i | c_{i-1})$ 是 c_{i-1} 出现后 c_i 的条件概率, $P(c_i | c_{i-2} c_{i-1})$ 是 $c_{i-2} c_{i-1}$ 出现后

c_i 的条件概率,以此类推。即使采用面向类的 n 元语法模型, n 的取值也不宜太大,常取 $n=2$ 或 $n=3$ 。

设语料库包含的词的总数为 N 。 c_i 的次数为 $f(c_i)$, $c_{i-1} c_i$ 和 $c_{i-2} c_{i-1} c_i$ 的次数分别为 $f(c_{i-1} c_i)$, $f(c_{i-2} c_{i-1} c_i)$, 则

$$P(c_i) = f(c_i) / N$$

$$P(c_i | c_{i-1}) = [f(c_{i-1} c_i) / N] / [f(c_{i-1}) / N] = f(c_{i-1} c_i) / f(c_{i-1})$$

$$P(c_i | c_{i-2} c_{i-1}) = f(c_{i-2} c_{i-1} c_i) / f(c_{i-2} c_{i-1})$$

即借助标注语料库可近似计算二元语法和三元语法的各种参数。

n 元语法配合隐马尔可夫模型,可求解一系列自然语言处理问题。任意一个句子的出现可看作是词随机排列的一个过程 $W = w_1 w_2 \cdots w_i \cdots w_n$,这种过程数学上叫马尔可夫过程。每个词有自己的词性,伴随词序列出现的词性序列 $C = c_1 c_2 \cdots c_i \cdots c_n$ 是一个“隐蔽的”随机过程。由于一个词的词性常不止一个,而一个句子只对应一个确定的词性序列。如何根据观察到的词序列确定隐蔽的词性序列,就是词性标注问题,其抽象的数学模型叫做“隐马尔可夫模型”。下面探讨其求解过程。根据联合概率和条件概率的定义,可以得到

$$P(C|W) = [P(C)P(W|C)] / P(W)$$

若对可能出现的不同的 C ,计算其条件概率 $P(C|W)$,其中使 $P(C|W)$ 取最大值的那个 C^* 就是所求的词性序列。即

$$C^* = \arg \max [P(C)P(W|C)] / P(W)$$

这里 \max 表示“最大值”, \arg 表示自变量, C^* 是使该函数取最大值的那个自变量。在这个问题中,序列 W 是不变的,上式等价于

$$C^* = \arg \max [P(C)P(W|C)]$$

剩下就是如何计算 $P(C)$ 和 $P(W|C)$ 。

对于 $P(C)$,可以采用二元语法,即

$$P(C) = P(c_1) P(c_2 | c_1) \cdots P(c_i | c_{i-1}) \cdots P(c_n | c_{n-1})$$

对于 $P(W|C)$,也要加以简化,即假定 w_i 只和相应位置上的 c_i 有关,于是得到

$$P(W|C) = P(w_1 | c_1) P(w_2 | c_2) \cdots P(w_i | c_i) \cdots P(w_n | c_n)$$

$P(c_1)$ 和 $P(c_2 | c_1), \cdots, P(c_i | c_{i-1}), \cdots, P(c_n | c_{n-1})$ 的计算公式已经给出。类似地, $P(w_i | c_i) = P(w_i, c_i) / P(c_i) = f(w_i, c_i) / f(c_i)$, $i=1, 2, \cdots, n$ 这里的 $f(c_i)$ 表示 c_i 的出现次数, $f(w_i, c_i)$ 表示 w_i

和 c_i 同时出现的次数。

于是根据一个规模足够大的语料库的词性频度 $f(c_i)$, 二元词性序列频度 $f(c_{i-1}c_i)$ 以及词 w_i 和词性 c_i 同时出现的频度 $f(w_i, c_i)$, 可以预测与一个词的随机序列 W 相伴出现的概率最大的词性序列 C 。用于训练 n 元语法和隐马尔可夫模型各种参数的语料库的建设通常需要程序自动加工和专家校对的协调配合。

参考文献

1. 冯志伟. 中国语料库研究的历史和现状——语料库研究回顾与问题. 见: Proceedings of ICC2001 (新加坡), 1~15
2. 俞士汶, 段慧明, 朱学锋, 孙斌. 北京大学现代汉语语料库基本加工规范. 中文信息学报, 2002, 16(5): 49~64, 16(6): 58~65
3. Teubert Wolfgang. Corpus Linguistics and Lexicography. International Journal of Computational Linguistics, 2001, 6(Special Issue): 125~153 (中文译文: 语料库语言学与词典学. 见: 靳光瑾编译. 计算语言学视窗. 北京: 北京广播学院出版社, 2003, 283~316) (俞士汶)

yuliaoku yuyanxue

语料库语言学 (corpus linguistics) 研究自然语言机读文本的采集、存储、标注、检索、统计等的语音学分支。目的是通过对客观存在的大规模真实文本中的语言事实进行定量分析, 支持语言研究和鲁棒的自然语言处理系统的开发。应用领域包括: 语言文字的计量分析、语言知识获取、作品风格分析、词典编纂、全文检索系统、自然语言理解系统以及机器翻译系统等。

现代语料库语言学的起源可追溯到 20 世纪 50 年代美国 Leonard Bloomfield 后期的结构主义语言学时代, 那时的语言学家在科学的实证主义和行为主义观点影响下, 认为语料库是一个规模足够大的语言数据库。这些语言数据是自然发生的, 因此对于语言学研究任务来说是必要和充分的, 而直觉证据充其量只是一种贫乏的第二位的资源。50 年代后期 Noam Chomsky 创建了转换生成语法, 他主张直觉是合理的, 而任何自然的语料都是扭曲的。他的这种理性主义的观点构成了当代理论语言学家的正统观念, 从而在极大程度上扼制了早期语料库语言学的发展。

促使语料库语言学在 20 世纪 80 年代再度崛起

的主要原因是:

(1) 30 年来自然语言处理系统的主流技术是基于规则的句法—语义分析, 赖以分析的语言知识无论是词典信息还是语法规则, 主要通过语言学家的内省来获取。但实践证明, 单纯依靠这样的知识不可能覆盖大规模真实文本中出现的各种语言事实。

(2) 计算机和计算技术的突飞猛进, 使语料库的规模从 20 世纪 60 年代的 100 万词次迅速增长到 90 年代的 (1~10) 亿词次, 30 年间扩大了上千倍。这是 50 年代拒绝直觉方法的早期语料库语言学家和 60 年代拒绝语料库方法的生成语言学家都未曾预见到的。这一事实使得一种语言的词汇、句法现象能够凭借语料库来进行开放性的调查。

1959 年有学者提出了建立现代英语用法调查语料库的设想, 20 世纪 60 年代初在美国布朗大学建立了现代美国英语的布朗语料库, 标志着语料库语言学第二个时期的开始。以键盘方式录入的布朗语料库和 70 年代创建的现代英国英语的 LOB 语料库被称为第一代语料库, 它们的库容量都是 100 万词次。80 年代, 光学字符识别 (OCR) 技术取代了语料的人工键盘录入方式, 使语料库规模迅速增长。这一时期建立的语料库有 COBUILD 语料库, 2 000 万词次; Longman/Lancaster 英语语料库, 3 000 万词次, 它们属于第二代语料库。进入 90 年代, 由于词处理编辑软件和桌面印刷系统的普及, 数量巨大的机读文本已成为语料库取之不竭的资源, 随之出现的是规模达 (1~10) 亿词次的第三代语料库。如美国计算语言学学会倡议的 ACL/DCI 语料库, 英国的牛津文本档案库等。根据 30 年来语料库规模的这种增长势头, G. Leech 预言公元 2021 年将出现 1 万亿词次的超大规模语料库。

诚然, 语料库的规模及其选材分布原则是重要的, 因为它们将直接影响到统计数据的可靠性和适用范围。但是作为语料库语言学研究的支撑环境来说, 语料的加工深度对语料库的作用关系更大。以汉语为例, 原始的“生”语料只能用来进行字频 (包括若干相邻字同现的频率) 和句长等统计, 提供简单的关键字检索 (KWIC)。为了实现词语一级的统计和检索, 就必须给原始语料加上分词标记。在后继的加工过程中, 还可以对语料进行词性、句法关系和义项等不同层次的标注, 使库存语料逐步由“生”变“熟”。随着语料所携带的各类信息日趋丰满, 语料库将最终成为名符其实的语言知

识库。

语料库语言学研究的主要内容包括:

(1) 基本语料库的建设;

(2) 语料加工工具的研究,包括自动分词系统、词性标注系统、句法分析系统、义项标注系统和话语分析系统等;

(3) 通过语料加工建立起各种带有标注信息的“熟”语料库;

(4) 从语料库中获取语言知识的技术与方法。

当前,世界上已建立了包括各种语言的几百个语料库,它们是各国研究人员从事语言研究和自然语言处理系统开发的重要资源。与此同时,有关语料库建设和利用的话题已成为国际学术刊物和会议的重要内容。在1993年7月召开的第四次机器翻译高层会议上英国学者所作的特邀报告中指出,自1989年起,全世界已进入第三代机器翻译系统的研究,其主要标志就是在基于规则的传统方法中引入了语料库方法,其中包括统计方法、基于实例的方法以及通过语料加工使语料库转化成语言知识库等。

参考文献

1. Garside R, Leech R, et al. The Computational Analysis of English: A Corpus-Based Approach. London: Longman, 1987

2. Aijmer K, Altenberg B. English Corpus Linguistics. London: Longman, 1991

3. 黄昌宁. 语料库语言学. 中国计算机用户, 1990, 11: 43 ~ 45 (黄昌宁)

yuyan chuli xitong

语言处理系统 (language processing system) 对软件语言进行处理的程序系统。

除了机器语言外,其他用任何软件语言书写的程序都不能直接在计算机上执行,都需要对它们进行适当的处理。语言处理系统的作用是把用软件语言书写的各种程序处理成可在计算机上执行的程序,或最终的计算结果,或其他中间形式。

不同级别的软件语言有不同的处理方法和处理过程。关于需求级、功能级、设计级和文档级软件语言的处理方法和处理过程是软件语言、软件工具和软件开发环境的重要研究内容之一。关于实现级语言即程序设计语言的处理方法和处理过程发展较早,技术较为成熟,其处理系统是基本软件系统之一。这里,语言处理系统仅针对程序设计语言的处

理而言。关于需求级、功能级、设计级和文档级语言的处理请参见需求定义语言,功能性语言,设计性语言,软件过程和软件工具。

程序设计语言处理系统随被处理的语言及其处理方法和处理过程的不同而异。不过,任何一个语言处理系统通常都包含有一个翻译程序,它把一种语言的程序翻译成等价的另一种语言的程序。被翻译的语言和程序分别称为源语言和源程序,翻译生成的语言 and 程序分别称为目标语言和目标程序。按照不同的源语言、目标语言和翻译处理方法,可把翻译程序分成若干种类。从汇编语言到机器语言的翻译程序称为汇编程序,从高级语言到机器语言或汇编语言的翻译程序称为编译程序。按源程序中指令或语句的动态执行顺序,逐条翻译并立即解释执行相应功能的处理程序称为解释程序。除了翻译程序外,语言处理系统通常还包括正文编辑程序、宏加工程序、连接编辑程序和装入程序等。

发展过程

随着程序设计语言的变化和发展,语言处理系统也跟着由小到大、由简单到复杂的变化和发展。最初人们直接用机器语言来描述问题的解法,这种程序无需任何处理就能直接在计算机上运行。但是这样的编程方式太繁琐,极易出错,效率极低,是非常不可取的。在计算机发展的早期,人们就在努力设法改变这种编程方式。开始时倾向于准备好一个由一些常用的例行程序组成的库,并借用一些代码来引用该库中的例行程序。后来改用一些字符或语言来表示这些代码,这样就成了符号语言的雏型。在此基础上,人们努力使机器语言符号化。机器语言发展成了汇编语言。语言的这一发展导致要求有一翻译程序把汇编语言程序翻译成机器语言程序,这种翻译程序称为汇编程序。

紧随汇编语言和汇编程序之后发展的是自动编译器。在自动编译器中,程序人员用的语言更接近通常的数学表示体系。但是用现在的标准来衡量,20世纪50年代初出现的第一批自动编译器都十分初步,它们只允许简单的单目运算,数据元素的命名方式有很多限制,然而它们促进了对高级语言处理系统和通用的翻译过程的研究。

20世纪50年代中期出现了FORTRAN等一批高级语言,与此相适应的语言处理程序、解释程序和编译程序也相继开发成功。

随着编译技术的进步和社会对编译程序需求的不断增长,50年代末有人开始研究编译程序的自动

生成工具,提出并研制**编译程序的编译程序**,它的功能是从任一语言的词法规则、语法规则和语义解释出发,自动产生该语言的编译程序。研制一个功能完全且实用的编译程序的编译程序是很困难的。多数编译程序的编译程序都是一些专用编译程序生成系统,如自动生成词法分析程序的扫描程序生成系统,自动生成语法分析程序的语法分析程序生成系统。

60年代起,不断有人开始使用自展技术来构造编译程序。自展的主要特征是用被编译的语言来书写该语言自身的编译程序。自展的思想最早在50年代中间就有人提出,到1971年,PASCAL的编译程序用自展技术生成后,其影响越来越大。

随着并行技术和并行语言的发展,处理并行语言的并行编译技术正在深入研究之中,将串行程序转换成并行程序的自动并行编译技术也正在深入研究之中。

分 类

按照处理方法,语言处理系统可分为编译型、解释型和混合型三类。

编译型语言处理系统是采用编译方法的语言处理系统。解释型语言处理系统是采用解释方法的语言处理系统。混合型语言处理系统是兼有编译和解释两种方法的语言处理系统。

多数高级语言都有一些不能在编译时刻确定而要运行时刻才能确定的特征。因此,与这些特征相关联的语言成分等价的目标代码在编译时刻不能全部生成,而要运行时刻才能全部生成。这些语言成分只能采用解释方法处理。多数解释程序都是先对源程序进行处理,把它转换成某种中间形式,然后对中间形式的代码进行解释,而不是直接对源程序进行解释。这就是说,多数高级语言处理系统既非纯编译型,也非纯解释型,而是编译和解释混合型。

基 本 内 容

程序设计语言处理系统主要包括正文编辑程序、宏加工程序、编译程序、汇编程序、解释程序、连接编辑程序、装入程序、编译程序的编译程序、自编译程序、交叉编译程序和并行编译程序等。

正文编辑程序用于创建和修改源程序正文文件。一个源程序正文可以编辑成一个文件,也可以分成多个模块编辑成若干个文件。用户可以使用各种编辑命令通过键盘、鼠标器等输入设备输入要编辑的元素或选择要编辑的文件,正文编辑程序根据

用户的编辑命令来创建正文文件,或对文件进行各种删除、修改、移动、复制及打印等操作。

宏加工程序把源程序中的宏指令扩展成等价的预先定义的指令序列。对源程序进行编译之前应先对源程序进行宏加工。

编译程序把用高级语言书写的程序翻译成等价的机器语言程序或汇编语言程序。编译过程可分为分析和综合两个部分。分析部分包括词法分析、语法分析和语义分析三步。分析的目的是检查源程序的语法和语义的正确性,并建立符号表、常数表和中间语言程序等数据对象。综合部分包括存储分配、代码优化和代码生成等步。综合的目的是为源程序中的常数、变量、数组等各种数据对象分配存储空间,并将分析的结果综合成可高效运行的目标程序。

汇编程序把用汇编语言书写的程序翻译成等价的机器语言程序。

解释程序按源程序中语句的动态执行顺序,从头开始,翻译一句执行一句,再翻译一句再执行一句,直至程序执行终止。和编译方法根本不同的是,解释方法是边翻译边执行,翻译和执行是交叉在一起的,而编译方法却把翻译和执行截然分开,先把源程序翻译成等价的机器语言程序,这段时间称为**编译时刻**,然后再执行翻译成的目标程序,这段时间称为**运行时刻**。正因为解释程序是边翻译边执行,所以要把源程序及其所处理的数据一起交给解释程序进行处理。

编译方法和解释方法各有优缺点。编译方法的最大优点是执行效率高,缺点是运行时不能与用户进行交互,因此比较适用于写规模较大或运行时间较长或要求运行效率较高的程序的语言,更适用于写机器或系统软件和支撑软件的语言。解释方法的优点是解释执行时能方便地实现与用户进行交互,缺点是执行效率低,因此比较适用于交互式语言。

连接编辑程序将多个分别编译或汇编过的目标程序段组合成一个完整的目标程序。组合成的目标程序可以是能直接执行的二进制程序,也可以是要再定位的二进制程序。

装入程序将保存在外存介质上的目标程序以适于执行的形式装入内存并启动执行。

编译程序的编译程序是产生编译程序的编译程序。它接受用某种适当的表示体系描述的某一语言类中任一语言A的词法规则、语法规则、语义规则

和(或)代码生成规则,并从这些描述产生出用目标语言 B 写的关于语言 A 的全部或部分编译程序。这样便可显著提高编译程序的开发效率。

自编译程序是用被编译的语言即源语言自身来书写的编译程序。利用自编译技术,可以从一具有自编译能力的语言 L 的一个足够小的子集 L_0 的编译程序出发,逐步构造出 L 的编译程序,也可从 L 的未优化的编译程序出发,构造优化的编译程序。

交叉编译程序是一种编译程序,它自身在甲机器上运行,生成的目标代码是乙机器的代码。

并行编译程序是并行语言的编译程序,或是将串行语言程序并行化的编译程序,后者又称为自动并行编译程序。

一个程序特别是中、大规模的程序难免没有错误。发现并排除源程序中的错误是语言处理系统的任务之一。通常源程序的语法错误和静态语义错误都是由编译程序或解释程序来发现的。排错能力的大小是评价编译程序和解释程序优劣的重要标志之一。源程序中的动态语义错误通常要借助于在语言中加入某些排错设施如跟踪、截断来发现和排除。处理排错设施的程序是**排错程序**。

展 望

语言处理系统的发展与软件语言、软件工程和软件技术的发展紧密相连,相互影响,相互促进。随着软件语言和软件技术向可视化、多媒体、并行化、智能化、自然化和自动化等方面发展,语言处理系统也向着这些方面发展。

参考文献

1. 徐家福. 系统程序设计语言. 北京: 科学出版社, 1983
2. Aho A V, Sethi R, Ullman J D. Compilers principles, Techniques and Tools. Addison-Wesley, 1986
3. Lowry M R. Software Engineering in the Twenty-First Century. AI Magazine Fall, 1992
4. 徐家福. 软件技术漫谈. 计算机科学, 1992, Vol. 19, No. 1 (徐永森 张素琴)

yuyi

语义(semantics) 语言成分的固有含义,亦即与言语情景(参见**语用**)无关的含义。在程序设计语言中,语言成分的语义就是该语言成分在程序执

行中应起之作用。语义研究涉及到的理论、原则、方法以及技术所形成的学科(或谓以语义为研究对象的学科)称为**语义学**。

语义具有如下基本特性:

固有性: 语义反映的含义是固有的,即与言语情景无关。

静态性: 语言成分的语义一般是在编译时刻可以确定的。

一元性: 语义可视为语言成分的一元函数。

语言成分的语义一般是用自然语言刻画的,用数学方法特别是用形式体系刻画的语义称为**形式语义**。随着刻画方法之不同,又可分为**操作语义**、**指称语义**、**公理语义**、**代数语义**等。

参考文献

Grice H P. Studies in the Ways of Words. Cambridge MA: Harvard University Press, 1991 (徐家福)

yuyi fenxi

语义分析(semantic analysis) 语言分析的一个分支,目的是根据上下文辨识一个多义词在指定句子中的确切意义,以及根据一个句子的句法结构和其中各词项的词义推导出这个句子的句义表达式。在**自然语言理解**和**机器翻译**中,用来表达句义的方式很多,常见的有:一阶谓词逻辑,语义网络,格框架(参见**格语法**)等等。语义分析的方法也会因采用的语义学理论和句义表达方式的不同而不同。例如,句子:

他给我三本书。

用一阶谓词逻辑来表示其句义时,可以定义一个三元谓词“给”:

给(x, y, z)

论元 x 代表“给”的动作发出者, y 表示受益者, z 表示给出的东西,那么这个例句的逻辑表达式为:

给(他, 我, 书(3))

其中,“书”本身是一个一元函数,其论元 3 表示数量。

上述例句句义的格框架表示如下:

(给:

施事(他),受益者(我),受事(书)),

(书:数量(3))

这个句子句义的语义网络表示,其实就是上述格框架的图形表示(见图 1)。

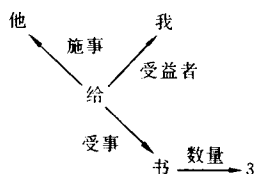


图1 句子的语义网络表示

参考文献

1. 石纯一, 黄昌宁, 王家祯. 人工智能原理. 北京: 清华大学出版社, 1993
2. 徐烈炯. 语义学. 北京: 语文出版社, 1990
(黄昌宁)

yuyi wangluo biaooshi

语义网络表示 (semantic network representation) 一种将知识中的对象与关系看作结点与有向弧, 并将其连成网状的表示形式。语义网络与**框架表示**是人工智能领域经常使用的两种表示方法。

1968年 Quillian 将英语词汇的语义关系表示为一种有向图结构, 并称其为语义网络。在这类图中, 结点与结点之间可以定义某种关系。例如, 折叠椅类是椅子类的一个子类, 一个具体的折叠椅是折叠椅类的一个实例, “子类”与“是一个实例”就表示了这些椅子之间的关系。进而, 后人根据语义网络的原理归纳出两种主要关系: 一种是“A是B的子类(A-Kind-Of, 缩写为AKO)”, 另一种为“A是B的一个实例(IS-A, 缩写为ISA)”, 这两种关系有一个重要的特性, 称为继承性。所谓继承性就是A将继承B的所有特性, 例如, 椅子有四条腿(B), 折叠椅是椅子的一个子类(A), 所以它将继承椅子的这个特性, 说明折叠椅也有四条腿。继承性是语义网络的最重要的性质, 这个思想是计算机科学中一种重要的程序设计方法——**面向对象程序设计的理论基础**之一。

一般一个语义网络由一些有向图表示的三元组(结点1, 弧, 结点2)连结而成, 其中弧是有方向的, 体现了主次关系, 结点1为主, 结点2为辅, 弧上的标注表示一个结点的属性或两个结点间的一种关系, 如ISA、A-Kind-Of(或Part Of)等。另外, 框架表示是语义网络表示一般化的表示方法, 两者没有本质上的区别。
(童颖 王珏)

yuyong

语用 (pragmatics) 语言成分相对言语情景的含义。言语情景包括发信人、收信人、语境、目标等。对**程序设计语言**来说, 语用乃语言成分在程序特定执行中的实际效用。语用研究涉及的理论、原则、方法以及技术所形成的学科(或称以语用为研究对象的学科)称为语用学。

语用具有如下基本特性:

相对性 语用反映的含义是相对言语情景而言的, 不是语言成分的固有含义。

动态性 语言成分相对言语情景而言的含义可视为语言成分的固有含义的实例, 它是动态生成的。

多元性 语用可视为语言成分与言语情景的二元函数。

今举二例如下:

例一. What did you mean by X?

这里反映的X的含义是语用, 它是X和you的二元函数。

例二. Ada 程序段。

```
pragma ..... ; Csi; .....
    opt ..... ; Csj; ..... Opt
pragma
```

其中 Csi, Csj 均为循环语句, 其含义和言语情景有关, Csi 除其循环语句的含义外, 还告知编译程序, 其目标码无需优化; 而 Csj 则除其循环语句的含义外, 还告知编译程序其目标码需优化。因而, 这里 Csi、Csj 的含义均为语用。

参考文献

- Geoffrey N L. Principles of pragmatics. Longman, 1983
(徐家福)

yu

域 (field) 一种具有两个二元运算的代数结构。

如果至少含有两个元素的集合 F 及其上的两个二元运算加法“+”和乘法“*”满足以下公理:

I_1 加法交换律 对任意 $a, b \in F$ 有 $a + b = b + a$;

I_2 加法结合律 对任意 $a, b, c \in F$ 有 $a + (b + c) = (a + b) + c$;

I_3 存在零元 存在 $0 \in F$, 使得对任意 $a \in F$ 有 $a + 0 = a$;

I_4 存在负元 对任意 $a \in F$, 有 $-a \in F$ 使 $a + (-a) = 0$;

II₁ 乘法交换律 对任意 $a, b \in F$ 有 $ab = ba$;

II₂ 乘法结合律 对任意 $a, b, c \in F$ 有 $a(bc) = (ab)c$;

II₃ 存在幺元 存在 $e \in F$, 使得对任意 $a \in F$ 有 $ae = a$;

II₄ 存在逆元 对任意 $a \in F$, 有 $a^{-1} \in F$ 使 $aa^{-1} = e$;

III 乘法对加法的分配律 对任意 $a, b, c \in F$ 有 $a(b+c) = ab+bc$ 和 $(a+b)c = ac+bc$ 。

则称 F 为一个域。域是交换环且无零因子。

全体有理数、实数和复数关于普通的加法和乘法都构成域,称为有理数域、实数域和复数域。

设 p 为一个素数。模 p 的全体剩余类 $\{0, 1, \dots, p-1\}$ 关于模 p 的加法 $+$ 和模 p 的乘法 $*$, 构成一个域,记为 $\mathbb{Z}/(p)$ 。

若 F 和 F' 为两个域,则环同态(同构) $f: F \rightarrow F'$ 就是域同态(同构) $f: F \rightarrow F'$ 。域 F 关于它的理想 D (即 D 是环 F 的理想)的商环 F/D 是域,当且仅当 D 是 F 的极大理想。

如果域 E 的子集 F 关于 E 的运算也构成域,则称 F 为 E 的子域, E 为 F 的扩域。有理数域是实数域的子域,实数域是有理数域的扩域。

设 F 为一个域。若对任意正整数 n 有 $ne \neq 0$, 则称 F 的特征为零;否则,有最小正整数 p 使 $pe = 0$, 这时称 F 的特征为 p 。显然 p 必是素数,而且对任意 $a, b \in F$ 有 $(a+b)^p = a^p + b^p$ 。

如果域 F 没有真子域即不同于 F 的子域,则称 F 为素域。若素域 F 的特征为零,则 F 与有理数域同构;若素域 F 的特征为 p ,则 F 与域 $\mathbb{Z}/(p)$ 同构。

研究域 E 的一般方法是取定 E 的一个子域(如 E 的素子域) F 作为基域,然后讨论 E 相对 F 的代数性质。对 E 的任意子集 M ,用 $F(M)$ 表示由 F 和 M 中元素经有理运算(加、减、乘、除)得到的所有元素的集合,则 $F(M)$ 既是 E 的子域,又是 F 的扩域,称为由 M 产生的 E, F 的中间域。若 $M = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$, 则 $F(M)$ 记为 $F(\alpha_1, \alpha_2, \dots, \alpha_n)$ 。称 E 的子域 $F(\alpha)$ 为 F 的单扩张。 F 的任意扩张都可以通过一系列单扩张得到。

设 E 是 F 的扩域。若 E 中每个元素都是 F 上某多项式的根,则称 E 为 F 的代数扩张,否则称 E 为 F 的超越扩张。如果 F 上每个多项式的根都在 E 中,则称 E 是 F 的代数闭域。有理数域的代数闭域是代数数域,实数域的代数闭域是复数域,域 $\mathbb{Z}/(p)$ 的代数闭域是多项式的分裂域。

如果 F 为有穷集,则称域 F 为有限域或伽罗瓦域。最简单的有限域是 $\mathbb{Z}/(p)$, 它的特征是 p 。取 $\mathbb{Z}/(p)$ 上一个 n 次不可约多项式 $f(x)$, 做 $f(x)$ 关于 $\mathbb{Z}/(p)$ 的分裂域即 $\mathbb{Z}/(p)$ 的包含 $f(x)$ 的所有根的最小代数扩域,这个域恰有 p^n 个元素。因为元素个数相同的有限域都互相同构,所以有限域的元素个数只能是 p^n 的形式,其中 p 为素数且 n 为正整数。具有 p^n 个元素的有限域记作 $GF(p^n)$ 。它是 $\mathbb{Z}/(p)$ 关于某个不可约多项式的分裂域。

域,特别是有限域,在理论上和算法上与计算机科学的许多领域密切相关,例如算法理论、复杂性理论、编码理论和机械定理证明等。另外,关于域和有限域的一些计算机方法的研究也在迅速发展。

参考文献

1. 万哲先编著. 代数和编码. 北京: 科学出版社, 1976
2. 熊全淹编著. 近世代数. 上海: 上海科技出版社, 1978
3. 王兵山, 周贤林, 王长英, 何自强编. 离散数学. 长沙: 国防科大出版社, 1985 (李廉 王兵山)

yuming xitong

域名系统 (domain name system, NDS)

Internet 中实现计算机名分级的机制。DNS 有两个概念上独立的要点: 一个是抽象的, 用来指明名字语法和名字的授权管理规则; 另一个是具体的, 用来指明一个分布式计算系统的层次结构, 并能高效地将名字映射到地址。

语法上, 每台计算机的域名由一系列字母和数字构成的段组成。例如, 清华大学计算机科学与技术系的域名为 cs.tsinghua.edu.cn。其中, cn 代表中国, edu 表示教育部门, tsinghua 表示清华大学, cs 代表计算机科学与技术系。域名是有层次的, 域名中最末一部分许多情况下都表示国家, 最左边的部分代表该台计算机的名字。

域名和 IP 地址(参见网际协议)是一一对应的, 域名易于记忆, 用得更普遍。当用户要和 Internet 上某台计算机交换信息时, 只需使用域名, 网络会自动转换成 IP 地址, 找到该台计算机。

域名系统规定了最高域的值, 如中国为 cn、日本为 jp、英国为 uk、美国的政府部门为 gov 等, 成为 DNS 的顶级域。在顶级域下的是二级域, 如 edu.cn 就是表示中国教育部门的域, 并可依次获得第三级域、第四级域……当一个组织希望参加域名系统时,

必须在某一级域下申请一个域名。一旦一个组织拥有一个已申请的域,就可以决定是否设置进一步的层次结构。对于一个规模小的组织可以不再设置下一级的域,而对于规模大的组织就有必要设置多层结构。

由于域名是一个逻辑概念,所以它与所处的物理位置可以没有关系。而且对一个组织来说有不同的部门,可以根据部门的实际情况选择不同层次的域名构造。

域名系统的一个特点是自治,即允许每个组织为计算机设置域名或改变这些域名。大多数具有 Internet 连接的组织都运行一个域名服务器,称 DNS 服务器。每当应用需要将域名翻译为 IP 地址时,应用成为域名系统的一个客户。这个客户将待翻译的域名放在一个 DNS 请求信息中,并将这个请求发给 DNS 服务器。服务器从请求中取出域名,将其翻译为对应的 IP 地址,然后,在一个回答信息中将结果地址返回给应用。

Internet 不采用集中式的命名机制,而采用分布式的名字空间管理机制,使名字和地址的映射任务分布执行。

名字空间的划分机制既要高效地支持名字映射,又要保证名字分配能自治控制。如果只考虑映射的效率,那么只需采用非等级名字空间,通过在多个映射机器之间划分名字来减少通信流量即可;如果只考虑管理的方便,则只需使管理机构的授权容易即可,但这样做会增加名字映射的开销和复杂性。

Internet 采用分级名字空间的管理,如同一个大的单位管理。在最高层划分名字空间,并指定代理负责下一级的名字管理。例如,一个名字空间的表示为 local. site, site 是由中心管理机构授权的网点名,local 是受 site 网点控制的名字的一部分,点是分隔它们的分界符。还可以进一步细分,例如,增加一个 group,名字空间表示为 group. local. site。原则上,可继续细分到足够小以便管理。

域名管理方案应包括一个高效、可靠、通用的分布式系统,以便实现名字对地址的映射。系统是分布式的,由分布在多个网点的一组服务器协同操作来解决映射问题;系统是高效的,大多数名字映射在本地操作,只有少数名字映射需要在互联网上通信;系统是通用的,因为它不限于仅使用机器名;系统是可靠的,单台计算机故障不会影响系统的正常运行。

域名对地址的映射由一组域名服务器完成。域名服务器中包含提供域名对地址转换,域名对 IP 地

址映射的服务程序。

图 1 是一个树结构的域名服务器的概念布局。树的根是能够识别顶级域,并知道如何解析每个域名的服务器。给定要解析的名字后,根可为该名字选择一个正确的服务器,并逐级往下搜索,最后将结果返回。

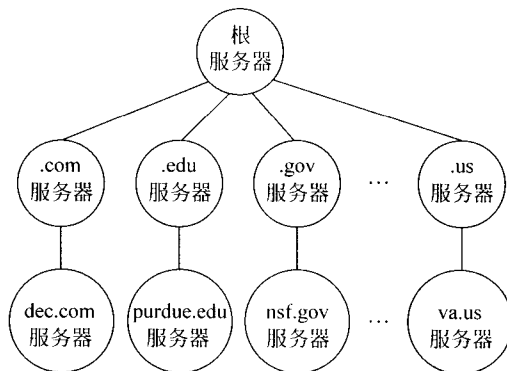


图 1 域名服务器树结构

概念树中的链接并不表示物理网的连接,而是解析域名的一种逻辑连接。服务器树是用于 Internet 通信的一种抽象结构。

从概念上讲,域名转换自上而下进行,从根服务器开始,逐级处理,直到树叶上的服务器。客户机必须知道如何与域名服务器联系,而域名服务器必须知道根服务器地址。域名服务器使用众所周知的协议端口通信,以便客户机方便地与域名服务器通信。

参考文献

1. 胡道元. 计算机网络(高级). 北京: 清华大学出版社, 1999
2. Douglas E Comer. Internetworking with TCP/IP, Volume I, 3rd ed. Prentice Hall Inc., 1995

(胡道元)

yuanqijian ceshi

元器件测试 (component testing) 用各种专用设备与有效方法,根据一定标准,对元器件进行的测试与选择。其目的是测试元器件的特性参数和性能,以及元器件筛选前后特性参数和性能的变化,从而剔除不符合技术条件要求的元器件,以提高整机系统的可靠性和稳定性。

集成电路测试以直流参数、交流或动态参数、静态及动态功能测试为主。直流参数用于测量电流、

电压和功耗等特性,交流参数或动态参数用于测量频率特性或脉冲特性,功能测试用于验证逻辑图形或真值表。静态功能测试在直流或低工作频率下进行,而动态功能测试则在最高工作频率下进行。线性集成电路测试以测量直流和交流参数为主。

晶体管测试通常以特性参数为主,测试项目有反向击穿电压、反向漏电流、电流放大倍数等。某些情况下,为了反映环境条件对特性参数的影响,还要进行高温或低温环境下的特性参数测试。

元件方面的测量主要是在不同的温度环境下进行静态参数的测试,如钽电解电容器通常在最高额定工作温度($85^{\circ}\text{C} \pm 2^{\circ}\text{C}$)下测试漏电流。

早期的元器件测试设备比较简单。随着电子计算机技术的飞速发展,元器件测试实现了测试过程、数据处理等自动化,测试精度也提高了。

参考文献

陈炳生. 电子可靠性工程. 北京: 国防工业出版社, 1987
(于海环)

yuanyqjian kekaoxing

元器件可靠性 (component reliability) 在元器件寿命期内和规定的条件下,保持其规定功能的能力。元器件可靠性分为固有可靠性和使用可靠性两类。

元器件固有可靠性由元器件的设计、材料和制造工艺等因素所决定。这些因素主要包括在批量生产中原材料选定、工艺条件设置、设备状况、人为因素的变动,以及线路、版面与结构设计的缺陷等。

元器件使用可靠性由元器件的选择与应用因素所决定。如在线路设计、布局和整机装配、调试中没能很好地考虑电和机械过应力对元器件的损伤,以及各种干扰和特性变化引起元器件误动作等因素都会影响元器件的可靠性。

元器件可靠性可用失效率 $\lambda(t)$ 来表征, $\lambda(t)$ 即产品工作 t 时刻之后的一个单位时间内失效的概率。大量的使用和试验结果表明,电子元器件典型的失效率曲线如图 1 所示。显见,失效率大致可划分为三个阶段,即早期失效期、偶然失效期和耗损失效期。

早期失效期出现在元器件开始工作后的较早期,其特点是失效率较高,且元器件失效率随时间的增加而迅速下降。早期失效是由于设计和制造工艺上的缺陷等因素造成的。可以通过加强对原材料和

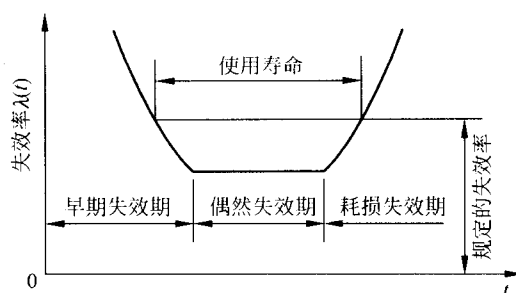


图 1 电子元器件典型的失效率曲线

工艺的检验,进行筛选等办法来淘汰早期失效的元器件,从而提高元器件可靠性。偶然失效期出现在早期失效之后,其特点是失效率低且稳定,近似常数。元器件的偶然失效期是元器件可靠工作的时期,研究这一时期的失效因素有重要意义。耗损失效期出现在元器件使用的后期,其特点是失效率随时间的增加而上升。耗损失效主要是由于元器件的老化、疲劳、损耗造成的。改善耗损失效的方法是不断提高元器件的工作寿命,对寿命短的元器件,在它们到达耗损失效期前就及时予以更换。为了提高元器件的可靠性,必须掌握元器件的失效规律,并采取有效措施。

参考文献

胡昌寿. 可靠性工程——设计、试验、分析、管理. 北京: 宇航出版社, 1988
(于海环)

yuanyqjian laohua

元器件老化 (component burn-in) 在较长的时间间隔内对元器件连续施加一定的电应力,通过电-热应力的综合作用来加速元器件内部的各种反应过程,促使其内部各种潜在缺陷及早暴露,从而达到剔除早期失效元器件的目的。

元器件老化是元器件筛选的一种形式。它对工艺制造过程中可能存在的一系列缺陷,如表面沾污、引线焊接不良、沟道漏电、硅片裂纹、氧化层缺陷、局部发热点、二次击穿等都有较好的筛选效果。对于无缺陷的元器件也可促使其电参数稳定。

一般计算机中需作老化的元件有电阻、电容等,器件有半导体晶体管和集成电路等。常用的老化设备有电阻、电容老化台,晶体管老化设备和微型计算机控制的模拟、数字电路老化系统等。

半导体器件常用的老化筛选方法如下。

常温静态功率老化 老化时,器件所处的环境温

度是室温,器件加正向偏压导通,老化所需要的热应力是由器件本身所消耗的功率转换而来。

高温静态功率老化 高温静态功率老化的加电方式及试验电路的形式均与常温静态功率老化的相同。区别在于前者在较高的环境温度下进行老化,集成电路的结温很高,因此,一般而言,前者的老化效果要比后者的好。

高温反偏老化 老化时,器件被同时加上高温环境应力和反向偏压电应力,器件内部无电流或只有微小的电流通过,几乎不消耗功率。这种老化方式对剔除具有表面效应缺陷的早期失效器件特别有效,因而在一些表面敏感的半导体器件中得到广泛应用。

高温动态老化 主要用于数字电路。其老化方法是,在高温环境下,向电路的输入端输入脉冲信号,使电路不停地处于翻转状态。这种老化方式比较接近电路的实际使用状态。

计算机所用的元件(电阻、电容等)主要采用高温静态功率老化的方法进行老化。

参考文献

胡昌寿. 可靠性工程——设计、试验、分析、管理. 北京: 宇航出版社, 1988 (于海环)

yuanqijian shaixuan

元器件筛选 (component screening) 将不符合使用要求的元器件通过目检或施加应力等方法使其缺陷暴露进而予以剔除的过程。从广义上讲,在元器件生产过程中各种工艺质量检验以及半成品、成品的电参数测试都是筛选。通常所提的计算机元器件筛选是指元件(电阻、电容等),器件(二极管、三极管、集成电路等)在安装使用前进行的筛选。筛选的方式如下。

目检筛选 根据被筛选元器件结构特点和目检要求,选用不同放大倍数的放大镜或显微镜,检查元器件的外观质量。目检筛选也可借助于红外显微镜、X射线仪等设备对元器件的内部质量进行检查。

温度应力筛选 在指定温度下进行。可分为以下三种情况:

(1) 高温存储 被筛选的元器件不加电应力,温度恒定。这项试验对有表面玷污缺陷的元器件有筛选作用。

(2) 高温电老化 被筛选的元器件加电应力,温度恒定。这项试验为具有加速性的筛选,它能暴

露元器件的潜在缺陷,从而把早期失效的元器件剔除。

(3) 温度循环 被筛选的元器件不加电应力,温度按一定规律,由低温突变为高温,随后又由高温突变为低温。这项试验对材料系数不匹配形成的缺陷或芯片已有裂纹的器件有筛选作用。

机械应力筛选 包括恒定加速度试验,振动试验(包括等幅振动、恒频振动、扫描振动、随机振动等)以及早期经常采用的跌落试验等。恒定加速度筛选试验对外壳封装不良,芯片黏结不牢,金丝内引线键合力不足等缺陷有筛选作用。

为了检查有空腔的元器件内有无可动多余物,可采用颗粒碰撞噪声检测(一种特殊的机械应力筛选)。

气密性筛选 采用浸在液体中检查是否漏气,放在真空环境中检查是否漏气等方法,其目的在于剔除达不到密封性要求的元器件。在选择筛选方法时要考虑元器件的封装形式和有效的内腔体积。

为了检查筛选前后元器件主要性能的变化,应对元器件的主要性能进行测试,测试的仪器设备应符合规定的精度要求。凡要求在筛选过程中进行监测的元器件,引线应尽可能短,测试过程不得产生寄生振荡。测试静电敏感器件时要采取防静电措施。

参考文献

朱明让. 质量与可靠性管理. 北京: 宇航出版社, 1988 (于海环)

yuan yuyan

元语言 (metalanguage) 用于描述另一语言的语言。在被描述的语言中不出现元语言的符号,元语言常用于描述程序设计语言的语法定义。例如,通常用巴克斯-诺尔形式体系(BNF)这种元语言来描述一般程序设计语言的语法。

元语言符号	意义
:: =	定义为
< >	生成项
	或

使用这些元语言符号,数字就可定义为:

< 数字 > :: = 0111213141516171819

无正负号整数可定义为:

< 无正负号整数 > :: = < 数字 > | < 无正负号整数 > < 数字 >

(程虎)

yuanshi digui hanshu

原始递归函数 (primitive recursive function)

一类特殊的可计算的数论函数。所谓数论函数是指：自变量值和函数值都是自然数的函数。所有原始递归函数都可通过有限次应用以下规则来定义：①函数值恒为0的零函数 C_0 、函数值为自变量值加1的后继函数 S 以及函数值为第 i 个自变量值的 n 元投影函数 $P_i^{(n)}$ 均是原始递归函数；②原始递归函数的合成仍是原始递归函数；③若 g 和 h 分别是 n 元和 $(n+2)$ 元原始递归函数，则由 g 和 h 按原始递归定义的 $(n+1)$ 元函数 f 也是原始递归函数。其中：

$$\begin{aligned} f(x_1, \dots, x_n, 0) &= g(x_1, \dots, x_n) \\ f(x_1, \dots, x_n, y+1) &= h(x_1, \dots, x_n, y, \\ &\quad f(x_1, \dots, x_n, y)) \end{aligned}$$

显然，所有原始递归函数都是直观可计算的全函数。许多常用的全函数都是原始递归函数，特别地，所有初等函数都是原始递归函数。但并非一切直观可计算的全函数都是原始递归函数。例如：阿克曼函数就不是原始递归函数。

原始递归函数具有以下重要性质：

(1) 若数论函数 f 的函数值只对自变量值的有限个组合非零，则 f 是原始递归函数。

(2) 若 g 是 $(n+1)$ 元原始递归函数，则由 g 按受限最小根定义的 $(n+1)$ 元函数 f 也是原始递归函数。其中：

$$f(x_1, \dots, x_n, y) = \mu' z [g(x_1, \dots, x_n, z) = 0]$$

即：若存在值 z 满足 $0 \leq z \leq y$ ，且使 $g(x_1, \dots, x_n, z) = 0$ ，则 $f(x_1, \dots, x_n, y)$ 等于使 $g(x_1, \dots, x_n, z) = 0$ 的最小 z 值，否则等于 $y+1$ 。

(3) 若 g_1, \dots, g_m 是 n 元原始递归函数，且 h_1, \dots, h_m 是 $(n+m+1)$ 元原始递归函数，则由 g_1, \dots, g_m 和 h_1, \dots, h_m 按联列递归定义的 $(n+1)$ 元函数 f_1, \dots, f_m 都是原始递归函数。其中：

$$\begin{aligned} f_1(x_1, \dots, x_n, 0) &= g_1(x_1, \dots, x_n) \\ &\vdots \\ f_m(x_1, \dots, x_n, 0) &= g_m(x_1, \dots, x_n) \\ f_1(x_1, \dots, x_n, y+1) &= h_1(x_1, \dots, x_n, y, \\ &\quad f_1(x_1, \dots, x_n, y), \dots, f_m(x_1, \dots, x_n, y)) \\ &\vdots \\ f_m(x_1, \dots, x_n, y+1) &= h_m(x_1, \dots, x_n, y, \\ &\quad f_1(x_1, \dots, x_n, y), \dots, f_m(x_1, \dots, x_n, y)) \end{aligned}$$

(4) 若 g 和 h 分别是 n 元和 $(n+2)$ 元原始递归函数，则由 g 和 h 按值程递归定义的 $(n+1)$ 元函数

f 也是原始递归函数。其中：

$$\begin{aligned} f(x_1, \dots, x_n, 0) &= g(x_1, \dots, x_n) \\ f(x_1, \dots, x_n, y+1) &= h(x_1, \dots, x_n, y, \langle f(x_1, \dots, x_n, 0), \dots, f(x_1, \dots, x_n, y) \rangle) \end{aligned}$$

这里 $\langle f(x_1, \dots, x_n, 0), \dots, f(x_1, \dots, x_n, y) \rangle$ 表示序列 $f(x_1, \dots, x_n, 0), \dots, f(x_1, \dots, x_n, y)$ 的哥德尔配数。

(5) 若 g_1, g_2 和 h 分别是 1 元、1 元和 4 元原始递归函数，则由 g_1, g_2 和 h 按双重递归定义的函数 f 也是原始递归函数。其中：

$$\begin{aligned} f(0, y_2) &= g_1(y_2) \\ f(y_1+1, 0) &= g_2(y_1) \\ f(y_1+1, y_2+1) &= h(y_1, y_2, f(y_1+1, y_2), \\ &\quad f(y_1, y_2+1)) \end{aligned}$$

(6) 有人已经证明：更一般的“范数序”意义下的函数递归定义方案也保持原始递归性。

原始递归函数和一类受限的编程语言之间存在对应关系。

参考文献

Hennie F. Introduction to Computability. Massachusetts: Addison - Wesley, 1977 (殷建平)

yuanying sucheng fangfa

原型速成方法 (rapid prototyping method)

一种快速建立预期系统的原型的软件开发方法。原型是预期系统的一个可执行模型，它正确地反映了系统性质的一个选定的子集（如功能、显示形式、计算结果或响应时间等）。原型可用来说明地表达和确认需求，验证所用设计方案的可行性和支持软件的演化。原型必须能快速、准确和低成本地构造和修改，并且是可运行的。

开发的原型可以在达到某种目的后被抛弃，也可以逐步将其演化成最终的软件产品。采用抛弃策略的原型主要用于目标软件的需求模糊不清的场合，或者难以确定设计方案可行性的场合。这类原型在明确了目标软件的需求或验证了设计方案后即被抛弃。由于原型最终被抛弃，所以，原型可以使用与目标软件不同的编程语言，可以运行于与目标软件不同的软硬件环境中。但是，它增加了对最终产品无用的原型开发费用和时间，因此原型的开发必须是快速的和低成本。同时这种方法还会诱使开发者省略或简化文档工作，这是有害的。抛弃策略是技术水平不够的一种权宜之计，最适合在项目早期阶段用来产生粗略的系统模型。

演化策略是在初始原型的基础上，通过不断扩

充和完善(每次扩充和完善产生一个新的原型版本),直至得到最终的软件产品。这类原型通常只提供目标软件的部分功能和性能,它们将是最终软件的一部分,因此,原型的准确规约和清晰的设计文档对有效的软件原型速成是至关重要的。

图1给出了一个演化的软件原型速成过程。首先对用户的要求进行需求分析,然后设计并产生初

始的原型,用户在观看了原型演示后,对原型作出评价,用户的反馈意见可能会引起新一轮的需求分析—设计原型—生成原型—演示原型的过程。原型在设计后,也可通过静态分析验证其合理性。原型的每个后续版本都应该更加接近于最终系统。被用户接受的原型经由一个可选的优化过程产生出发布的产品。

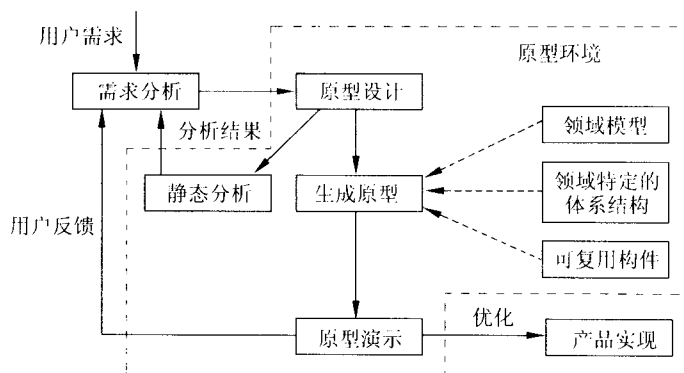


图1 原型速成过程

有时可以针对所需软件的各种问题开发多个小的原型,每个小原型回答一个问题,这会更有用。例如,针对用户界面的界面原型,针对某些特定功能的功能原型等。这种做法的好处在于反映系统不同方面的原型可以并行独立开发,每个原型都相对比较小、简单、容易改变。

原型速成方法已经被广泛接受。**演化模型、螺旋模型**和快速应用开发(RAD)等都是基于原型速成方法的。虽然开发原型需要花费一定的成本,但由于原型有利于明确需求和确定设计方案,因此采用原型速成方法能避免因需求不清或设计方案不合理而造成的损失。

参考文献

Marciniak John J. Encyclopedia of software engineering. John Wiley & Sons Inc., 2002 (钱乐秋)

yuancheng jiaoyu

远程教育 (distance education, distance learning) 函授教育、广播电视教育、网络远程教育等时空相隔的师生之间通过一定的媒介进行的有组织的教学活动。

远程教育起源于19世纪中期,大体上经历了三个发展阶段。早期是通过信函传送教学内容,学校

将教学材料邮寄给学生,学生独立地学习,在学习过程中,学生和教师可通过信函进行信息交流,这种形式的远程教育称为函授教育(学)。随着广播、电视的出现和普及,教学内容通过广播或电视媒体播放出去,学生可以通过收听广播或收看电视学习,在学习过程中学生和教师还是通过信函或电话交流信息,因此,称为广播电视教育(学)。20世纪80年代,全球互连网络 **Internet** 的出现,人们转而通过**计算机网络**传送教学内容,学生可以通过 **Internet** 访问网络上的教学信息。另一方面,Internet 应用的不断丰富,**万维网**、**电子邮件**、多媒体会议系统等应用系统的出现,使学生与教师之间、学生与学生之间以及教师与教师之间可以进行同步或异步的信息交互。计算机网络技术的发展使远程教育进入了一个全新的发展阶段即网络远程教育。未来的教育将向着社会化、全球化的方向发展,它将给人们提供开放的、平等的、协作的和竞争的学习和工作空间,人们的工作和学习不再是截然分开的两个阶段,而是密切结合在一起,贯穿于人的一生,这也就是真正意义上的“活到老,学到老”。

网络远程教育的特点

基于网络的远程教育与传统的远程教育相比具有以下特点:

- (1) 访问方式的时空无限性
- (2) 教学信息的共享性
- (3) 教学方式的双向交互性
- (4) 学习模式的多样性

网络远程教学的形式

目前,网络远程教学已出现了以下形式:

(1) 远程访问 通过联机访问远程信息资源进行学习,如访问电子图书馆、数据库、博物馆、卫星数据以及远程教室。

(2) 远程体验 借助于通信网络和其他电化教育手段,学生可以从远程来体验和感受一些事件的发生过程和结果,这些经验在传统的教学环境中只能间接地获得。

(3) 远程辅导 学生可通过电子公告牌、新闻组、电子论坛等多种交流和辅导形式获得专家的辅导。这种远程辅导形式对于教师进修和成人教育有积极意义。

(4) 远程共享 这种源于电子邮件系统的远程教育方式可推动学生之间的交流和教师之间的合作,共享数据、经验、主意、发现等信息系统。这种教育模式将改变传统垂直结构的师生关系,提高参与者的平等性。

(5) 虚拟出版 借助于万维网等工具,信息的发表不再受印刷工具的限制,丰富了信息的表现形

式,同时也增强了学生的学习积极性。

(6) 虚拟教室 虚拟教室是指在计算机网络上利用虚拟现实技术和多媒体技术构造的虚拟学习环境,允许位于异地的教师和学生互相感知。不但可以利用实时通信功能实现物理教室中所能进行的大多数教学活动,还能利用异步通信功能实现前所未有的教学活动,如异步辅导、异步讨论等。

(7) 计算机支持的协作学习(CSCL) 是在计算机网络技术环境支持下,学习参与者可突破地域和时间上的限制,进行相互交流、信息共享和合作性学习,又称为教育群件系统。为学习参与者提供同步和异步的多媒体信息服务和共享学习空间。它具有以下两个特点:主动的交互活动和群体的协作学习。

网络远程教学软件模型和教学运行系统

远程教学软件的设计,没有一个固定的模式可以遵循,应该根据不同的需求设计不同的教学软件。一个完整的远程教学系统应该包括两个部分:课件开发系统和教学运行系统。课件开发系统为课件开发者提供一个开发环境;教学运行系统为学习参与者提供一个学习环境,如图1、图2所示。

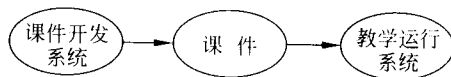


图1 一个完整的远程教学系统

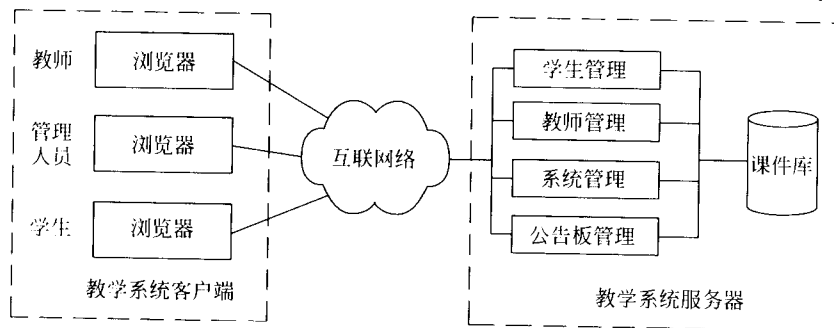


图2 一个基于Web的教学运行系统

基于网络的远程教学系统常常采用**客户-服务器模式**。服务器主要提供学生管理、教师管理、系统管理和教学信息维护等功能。客户端运行万维网浏览器,它为学生、教师以及管理人员使用远程教学系统提供一个集成环境和用户接口。

参考文献

1. 史美林. 计算机支持的协同工作: 理论和应用. 北京: 电子工业出版社, 2000

2. Watabe K, Hamalainen M & Whinston A B. An Internet Based Collaborative Distance Learning System: CODILESS. Computer and Education, 1995, 24 (3)

(史美林)

yuancheng wangluo jiankong

远程网络监控(remote network monitoring, RMON) 一种用增强简单网络管理协议

(SNMP)功能来实现网络远程监视的过程和技术。网络管理站通过 SNMP 与代理通信以访问管理信息库(MIB)。SNMP 在轮询的过程中,会产生大量的通信,给网络带来很重的负载。有了远程网络监控(RMON)后,在一个局域网(LAN)或一个网段上只要设立一个网络监视器,对网络数据进行监控、收集、分析和存储,并对网络状况不断进行测试,将收集到的信息记录反馈给网络管理站,可大大降低网络的开销。RMON 在执行和诊断功能上也对 SNMP 有所扩展,它不但在局域网中可有效地改善性能,在广域网环境中也显示出其巨大的优点。

Internet 工程特别工作组 IETF 于 1991 年发布了征求意见文献 RFC 1271,即远程网络监控管理信息库(RMON MIB),后来在 RFC 1513 中又定义了针对**权标环网**的远程网络监控标准。在 RFC 1271 和 RFC 1513 中,远程网络监控的设计目标被确定为:实时监视、故障检测、脱机操作、数据收集、多管理支持以及图形化显示。

参考文献

Stallings W. SNMP, SNMPv2 and RMON. MA: Addison - Wesley, 1996 (高传善)

yuancheng yiliao

远程医疗 (telemedicine) 在计算机网络环境下开展的异地远程医疗活动。特别是在 Internet 环境下,在**医疗管理信息系统**的基础上,异地开展的远程医疗咨询与诊断、远程专家会诊、在线检查、远程手术指导、医疗信息服务、远程教学和培训等活动,乃至建立一家基于网络环境的虚拟医院。远程医疗也称为远程医学。

远程医疗的开展应该有国家卫生部门的统一规划,制定法规与制度,统一医疗信息标准,授权建立不同级别层次的远程医疗管理中心,严格审查会诊医院和医生的资格并建立相应的档案,严格收费标准,保障通信与医疗设施。建立的远程医疗系统应具有开放性、可用性、先进性、经济性、安全性和用户友善性、法规与法律保障性。

远程医疗系统的组成

远程医疗系统的组成可分为四个层次,如图 1 所示。

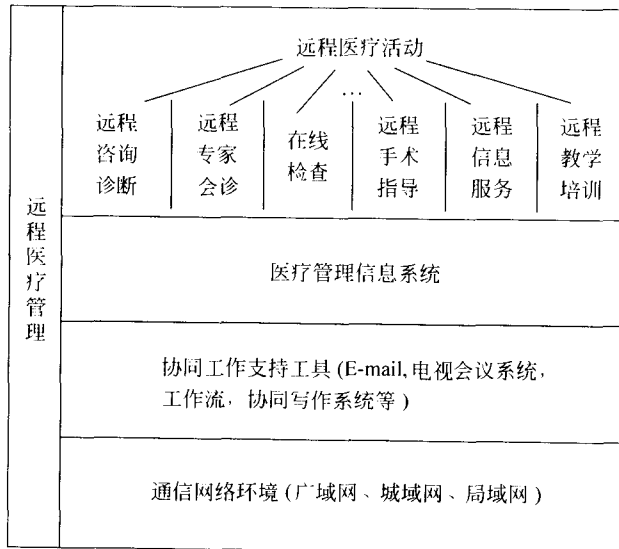


图 1 远程医疗系统体系结构

基于远程专家会诊的远程医疗系统的一般组成原理如图 2 所示。它由请求会诊的医院之医生及病人、参与会诊的医院的专家医生(可能不止一个医院的专家)、会诊管理中心等单位通过专用网或公

用网,特别是 Internet,形成一个远程医疗网络系统。有关的这些医院是该系统的会员单位,他们围绕病人在会诊管理中心的协调下开展协同医疗活动。

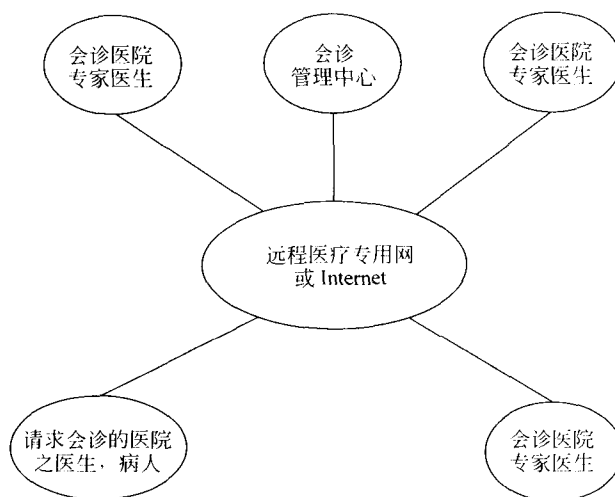


图2 远程医疗系统组成原理

远程医疗系统的工作模式或服务模式可以分为异步非实时和同步实时两类。前者通过电子邮件(信函)进行医疗咨询或会诊。后者往往通过电视会议系统进行远程实时会诊乃至手术指导等医疗活动。

远程医疗的协同工作组织问题

远程医疗协同工作有五个不同的协作层次,即数据通信、信息通信、协调、合作和协同。如何组织好这五个不同层次的协作是有效实施远程医疗的关键。通过对协同决策系统的分析,我们可以给出以下6个远程协同医疗中的要素:①决策者——医生;②决策对象与目标——病人疾病;③决策数据——病人病情数据信息;④通信环境;⑤决策生命期——诊断过程;⑥决策实现——医生会诊并提供诊断报告。合理地组织好这些要素并建立相应的关系才能有效地完成一次远程协同医疗。

(1) 远程医疗通信网

远程协同医疗模式及要素中,通信环境是保证正确的远程传输病人的医疗数据的基础,特别是多媒体医学图像的正确传输尤为重要。为了开展基于多媒体电视会议系统的实时远程会诊乃至手术指导,建立高速宽带的多媒体信息网是十分必要的。而在以E-mail、万维网等方式进行的远程医疗文件的提交与检索、异步非实时的咨询、会诊模式中,相对地讲对网络带宽要求可以低一些。因此,根据开展远程医疗的模式的不同,可建立相应的虚拟专网,如租用公用的宽带网、卫星网、Internet网等构建远

程医疗通信基础设施。

(2) 远程医疗中的信息协同

参与远程协同医疗的各成员单位(包括会诊中心、请求会诊医院、会诊医院、医疗保险机构等)正确互通与理解医疗数据信息是远程医疗中的协同的第二个层次,即信息协同。由于各成员单位的医疗信息系统不一定是统一开发的,医疗信息表示、格式、数据标准等不尽相同,这会给协同医疗带来困难。因此从信息协同的角度考虑,远程医疗系统开始建设时,在远程医疗管理中心的统一管理协调下,有统一的各成员的远程医疗信息分系统是十分必要的,至少在远程医疗管理中心要有对各成员单位有相应的信息转换接口、统一的标准,以保证医疗信息的协同。如果在一开始就能规划与研制面向远程医疗的“协同数据库”或“数据库协同管理系统”那对于今后远程协同医疗将会带来很大方便。

(3) 远程医疗中的协调、合作和协同

病人及其病情是远程医疗的对象(客体)。请求会诊医院及医生、会诊医院及医生是远程医疗中的主体,他们要围绕病人病情开展咨询、讨论、诊断。会诊管理中心则负责医疗全过程的组织与协调。主、客体在组织者管理下按照一定的医疗程序开展协同医疗活动。图3给出了两种远程医疗工作模式的组织协调过程。

参考文献

1. 史美林. 计算机支持的协同工作: 理论和应用. 电子工业出版社, 2000

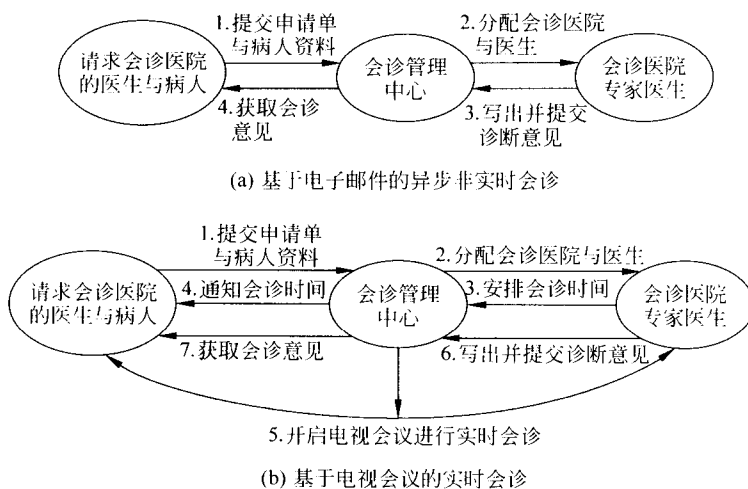


图3 两种远程医疗工作模式的协调过程

2. 中华医学会. 中国远程医疗网 (<http://www.telmedicine.com.cn>) (史美林)

yueshu tuili

约束推理 (constraint reasoning) 一种基于关系表达式的一致性推理。含有变量的关系表达式称为约束。约束给定了问题的有关限制。约束推理的目标就是求满足所有这些限制的解,因而约束推理也称约束满足问题。在一般情况下,这种求解需要在状态空间中搜索。约束推理研究的内容主要包括各种有效的搜索策略及约束的表示。

约束推理最早起源于计算机景物分析的研究,约束用来定义对图元(如线段)的一致解释。为了解决解释的歧义性问题,D. Waltz 提出了基于弧一致性的约束传播技术。这种技术后来发展成为著名的弧一致算法 AC 3 及其多种推广形式。R. Stallman 与 G. J. Sussman 用约束来表达电路与机械设备的功能与结构,并实现了依赖制导的回溯技术。约束传播与智能回溯成为后来约束推理的主流技术。

约束满足问题可形式地定义如下:

给定: ①变量的一个有限集合 V , 每个变量都对应一个离散有限集合,称为该变量的域; ②关于 V 中变量的约束的一个有限集合 C , 每个约束定义于 V 的有限子集并限定该子集中变量由其域中取值的组合。

求: 对 V 中所有变量的一个(或多个)赋值,使得 C 中所有约束都得到满足。

如果所有的约束都是二元或一元的,则变量间的约束相关关系称为约束满足问题的约束图。

求解约束满足问题的传统方法有:

(1) **回溯法** 依次对各个变量赋值,并及时对各约束求值,每当发现约束不满足时,回溯到一个已赋值的变量,重新选择其值。其中最简单的有顺序回溯法;较复杂的有各种智能回溯技术,包括回跳法及真值维护系统。

(2) **约束传播** 在约束图上传播变量取值的限制,主要有值的传播与域的传播。最常用的技术是弧一致性算法。

(3) **动态确定变量的赋值次序** 其中一个最有效的启发式是选择当前域最小的变量赋值。

这些技术的研究至今仍很活跃。此外,最新的研究方向还包括:

(1) **结构启发式** 研究约束图的拓扑结构及其在启发式搜索中的作用;

(2) **默认约束及软约束的求解**,如部分约束满足;

(3) **修正法搜索** 运用爬山法技术,通过不断修正各个变量的取值,逐渐减少违反约束的个数,直至为零;

(4) **连结主义搜索** 运用神经网络技术求解约束满足问题;

(5) 将上述技术结合起来的集成技术。

约束表示广泛地应用于人工智能的各个领域,包括定性推理、基于模型的诊断、自然语言理解、景

物分析、任务调度、系统配置、科学实验规划、机械与电子设备的设计与分析等。

参考文献

1. Dechter R. Enhancement Schemes for Constraint Processing: Backjumping, Learning, and Cutset Decomposition. *Artificial Intelligence*, 1990, 41(3): 273 ~ 312
2. Kumar V. Algorithms for Constraint Satisfaction Problems: a Survey. *AI magazine*, 1992, 13(1): 32 ~ 44
(史忠植 廖乐健)

yundong fenxi

运动分析 (motion analysis) 以不同时刻所得到的多幅图像为数据,推导所观察的运动物体的三维形状及运动速度的方法,是计算机视觉的一个重要研究方向。在一些文献中,该研究方向也称为“由运动分析物体结构”或序列图像分析。对人类视觉功能的分析表明,人具有从一系列二维图像估计运动物体的形状与运动速度的能力。

图像分割是图像处理中的一个困难的问题,在单幅图像中,由于物体与物体之间,物体与背景之间互相遮挡,要把单个物体与背景所属的图像区域分割开,尤为困难。但利用运动分析方法,并假设各个物体在做不同的刚体运动,就可以较容易地进行图像分割。

运动分析的另一个重要应用是序列图像压缩,利用运动分析将图像中的运动物体所对应的区域分割出来,并只对这部分进行编码,可大大提高序列图像的压缩比。

运动图像的分析方法可归结为两类,一类称为**基于特征的方法**;另一类称为**光流法**。

基于特征的方法分三步计算物体的三维运动和形状。第一步检测各图像上的特征点或特征线(例如多面体的顶点或棱线);第二步确定这些特征在各图像间的对应关系,这些对应关系给出空间物体表面的特征点(或线)在图像上的投影位置随时间的变化;第三步是在刚体运动的假设下,由以上对应关系计算物体的形状与运动速度。

由运动物体表面的点在图像平面上投影的二维运动场(称为光流场)进行运动分析的方法称为**光流法**。分析表明,图像上每一点的灰度沿空间与时间方向的微分,提供了对上述二维运动场的约束条件。由这些约束条件及物体表面各点在空间上的连续性约束,可计算出光流场。然后,由光流场与三维

物体的刚体运动假设,可求出三维物体的形状与运动速度。

基于特征的方法中,最大的困难是对应关系的确定,而光流法涉及到计算图像灰度的一次甚至二次导数,因而对噪声的影响非常敏感。上述两种方法所得到的解,都不是惟一的,存在多解问题,因而,还需要一些其他的知识(如三维物体形状的一些先验知识),才能确定惟一的解。

除特殊情况外,上述运动分析的方法,都涉及到求解复杂的非线性方程组,因此,无论是从计算理论还是从计算方法的角度看,运动分析还是计算机视觉中的一个研究方向,目前还没有达到实用化的阶段。

由于运动分析有极其广泛的应用前景,如视觉导航、机器人装配、运动目标跟踪、医学诊断等方面。因此,运动分析目前仍是计算机视觉最为活跃的研究方向之一。

参考文献

1. Huang T, Tsai R. Image Sequence Analysis: Motion Estimation. In: *Image Sequence Analysis*, ed. by T. Tuang, Springer Ser. Info Sci., Vol. 5. Berlin, Heidelberg: Springer, 1981
2. Nagel H. Image Sequences-ten years, from Phenomenology Toward a Theoretical Foundation. *Proc. 8th Int. Conf. Pattern Recognition*, Paris, France, 1986, 1174 ~ 1185
(马颂德)

yundong tuxiang de yasuo bianma
biaozhun

运动图像的压缩编码标准 (compression and coding standards for motion image) 对内容随时间变化的图像序列(又称动态图像)进行压缩编码的技术标准。

运动图像实时地记录了景物的动态变化过程,它需要 25 ~ 30 帧每秒图像来表示,数据量十分巨大。由于序列中帧与帧之间画面内容存在高度的相关性,变化仅发生在局部范围内。如果把图像中的变化部分用运动向量来描述,那么某一帧的图像就可以看成它的前帧图像经过运动向量补偿后的结果。另外,两帧图像只要时间间隔不长,它们的中间帧图像的变化基本上是该两帧图像的平均变化,即两帧图像的插值。因此,可以把序列中的图像分为帧内图、前向预测图、双向预测图 3 种类型。帧内图

的压缩采用静止图像压缩技术(参见**图像的压缩编码**),以减少空域冗余度。前向预测图用前面的帧内图根据运动向量进行预测补偿。双向预测图可以根据前面和后面的图像进行前向、后向或者双向预测。由此可见,运动图像仅需要使用帧内图和运动向量表示,因而有相当高的压缩率。

运动图像压缩编码的国际标准主要有两个系列:国际标准化组织(ISO)建议的 MPEG 标准和国际电信联盟 ITU 的 H. 26x 标准。

H. 261 第一个被广泛使用于视频会议的视频编码标准,由 ITU-T 在 1991 正式通过。它的码率为 80 ~ 320 kb/s。1993 年又通过了第二版,目标码率为 64 ~ 2 048 kb/s。

MPEG-1(IS-11172) 已广泛使用于 VCD 等的音频和视频编码标准,质量与 VHS 类似或者稍好,码率为 1.5 Mb/s。它是由 ISO/IEC JTC1 MPEG 运动图像专家组在 1993 年制定的。其视频编码的性能比 H. 261 要好。

MPEG-2(IS-13818)/**H. 262** 继 MPEG-1 之后又一个被广泛使用于 DVD 和数字电视等领域的音频和视频编码标准,与 MPEG-1 相比,图像质量要高许多,码率为 4 ~ 30 Mb/s。它是由 ISO/IEC 和 ITU-T 在 1994 年制定的。与 MPEG-1 不同的是,MPEG-2 系统层已经考虑到了传输,有节目流和传输流两种。其中节目流用于误码率低的应用环境中,它只包括一个节目,而传输流可用于误码率较高的传输环境,可以同时包含多个节目。

H. 263 超低码率的视频编码标准,在超低码率带宽(低于 30 kb/s)下其技术比较先进,与 H. 261 相比,获得相同的图像质量时码率仅为二分之一。最初

预定码率范围是 10 ~ 30 kb/s,在制定过程中扩展到 10 ~ 2 048 kb/s。ITU-T 在 1996 年通过了 H. 263 的 1.0 版本,此后又分别制定了技术更加先进的 H. 263 + 和 H. 263 ++ 两个版本。

MPEG-4(IS-14496) 用于交互式多媒体通信的音频和视频编码标准,1999 年和 2000 年分别通过了 1.0 和 2.0 版本,目前还正在制定 V3.0 和 V4.0 两个版本,其中 V3.0 支持演播室级分辨率的高码率编码,而 V4.0 则支持面向异构网络传输的高可靠交互功能。MPEG-4 视频编码标准除包括此前所有标准的编码方法(包括静止图像编码方法)之外,还新增加了一些新的功能,如对静态纹理的小波编码,任意形状区域的编码,合成和自然视频的混合编码等。它可支持的码率范围、图像分辨率和帧率范围很宽,并支持交织扫描格式。低码率时其效率与 H. 263 相当,而高码率时与 MPEG-2 相当。MPEG-4 还针对无线网络传输增加了容错和可伸缩编码的功能。该标准将是下一代网络应用的主流多媒体通信标准。

H. 264(MPEG-4 AVC) 用于多媒体通信的视频编码国际标准,由 ISO/IEC MPEG 和 ITU-T 高级编码专家组 VCEG(SG16/Q15) 共同制定。其目标是在各种码率条件下,都要显著提高视频编码的效率。它采取了多种先进技术,例如采用多种不同大小块的运动补偿和 1/4 像素精度的运动向量,并针对低码率下的块效应采用了环内低频滤波,以减少块效应的影响等。

上述几种视频压缩编码国际标准的发布时间、码率范围和相应应用领域可参见图 1。

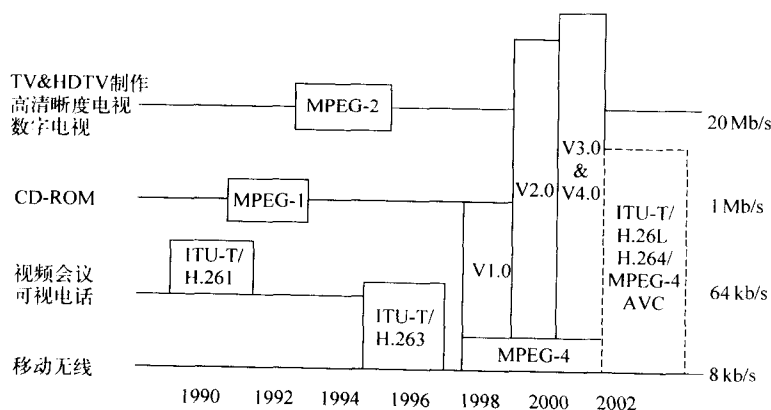


图 1 视频压缩编码的国际标准

参考文献

1. 钟玉琢,王琪,赵黎,杨小勤编译. MPEG-2 运动图像压缩编码国际标准及 MPEG 的新进展. 北京:清华大学出版社,2002

2. 钟玉琢,王琪,贺玉文. 基于对象的多媒体数据压缩编码国际标准——MPEG-4 及其校验模型. 北京:科学出版社,2000 (贺玉文)

yunsuanqi

运算器 (arithmetic unit) 参见中央处理器。

yunsuan sudu pingjia

运算速度评价 (arithmetic speed evaluation) 采用计算分析、模拟、测试等方法 and 工具,对计算机系统执行算术逻辑运算的速度的推断和测定。运算速度通常用每秒执行指令的条数来表示。

计算机系统运算速度是系统工作能力和生产效率的主要表征,是设计者和使用者共同关注的重要性能指标。

早期的计算机系统以算术运算为主要操作,常用加法指令的执行速度来表征机器运算速度。但是,由于计算机系统日趋复杂,仅用加法指令已难以评估机器的运算速度。1959 年,吉布森提出从应用课题程序中统计各类指令所占百分比,用指令混合比计算指令平均执行速度,称吉布森混合法。对不同应用领域,由于程序类型不同,所用指令混合比也不一样,其比值由所在领域大量应用程序统计归纳得出。人们常采用 MIPS(百万[条]指令每秒)和 MFLOPS 值(百万[次]浮点运算每秒)来表征计算机的平均运算速度,即让计算机执行按一定比例混合的各种基准程序,计算出整数平均运算速度和浮点平均运算速度。由于系统运算速度与很多因素有关,生产厂家宣称的运算速度与用户实际观察到的差距很大,于是,人们设法模拟用户的实际负载,抽出常用算法,进而编制通用测试程序,用以测试不同类型、不同型号的计算机系统的运算速度,这种程序称为**基准程序**。

决定计算机系统运算速度的因素很多,各因素影响的程度也不同,主要因素有:

(1) 处理器硬件技术指标 决定处理器运算速度的硬件技术指标越高,运算速度越快。如:处理器主时钟频率、字长、寄存器数量、高速缓存和主存容量与存储周期等。

(2) 处理器体系结构 采用先进的处理器体系结构,可以降低每条指令执行的平均节拍,使运算速度提高很大。如 RISC 技术,采用超标量、超流水线、超长指令字的多发射结构处理器,可以达到每个机器节拍执行多条指令。

(3) 并行处理体系结构 指令流和数据流的并行处理,使计算机系统运算速度迅猛增长。如:多功能部件、向量处理器、多处理器系统、大规模并行处理系统等等。

(4) 应用课题程序的适应特性 实用课题程序内相关特性,常常制约系统运算速度的发挥,导致实际运算速度下降。主要原因是:①程序数据局部性差,使高速缓存命中率低,争用总线和访问主存延迟增大;②指令或数据相关性强,使得操作流水线频繁中断;③向量化率低的课题程序,使得向量处理器的流水线空闲;④程序和数据可分割性差,使得大规模并行处理部件难以发挥作用。

运算速度评价方法很多,主要有:

(1) 理论计算方法

单处理器运算速度 包括:①计算各种指令执行速度。根据单处理器主时钟频率 f (MHz),求出单处理器基本工作节拍 T , $T = \frac{1}{f}$ (ns)。再根据处理器结构模型和指令操作流程,推算出执行各种指令的基本节拍和每秒执行指令的次数。②计算吉布森混合比运算速度。假设第 i 类指令 ($i = 1, 2, \dots, n$) 在使用过程中的概率为 P_i ,其执行时间为 t_i ,则平均执行指令时间 t_E 为

$$t_E = \sum_{i=1}^n P_i t_i$$

t_E 的倒数即吉布森混合比平均运算速度,其中 P_i 的集合即为混合比。③计算峰值运算速度。假设指令和数据都在高速缓存内,全部流水线无阻塞、全部功能部件都满载工作的最佳环境下,计算出处理器理论设计的最高运算速度。④计算核心程序运算速度。经过实际统计,一个较大程序,占程序总运行时间 90% 以上的“核心”,只有为数不多的指令条数,这种程序“核”称核心程序。计算核心程序的运算速度,能反映机器在该应用领域较为实际的运算速度。⑤计算典型程序运算速度。选取实际应用中有代表性的课题,如:FFT,图形图像处理等,计算出典型程序运算速度。此外,还有所谓“数据处理速率”(PDR)和“综合理论性能”(CTP)值等,这些都是美国政府限制高档计算机出口而制定的运算速

度估算方法。

系统运算速度 在计算单处理器运算速度的基础上,计算系统运算速度,包括:①计算系统峰值运算速度。若单处理器峰值运算速度为 R_{peak} ,系统内处理器的数量为 N ,则系统峰值运算速度为 R_{peak} 乘以 N 。有时也用单处理器平均运算速度 R_{mean} 乘以 N 来表征系统峰值运算速度。②计算系统平均运算速度。运用模型分析和模拟方法,可以计算较为近似的系统平均运算速度。

模型分析方法 可为计算机建立一种用数学方程表示的模型,进而在给定不同参数的条件下,分析和计算影响系统运算速度的因素。计算机系统由一组有限的资源组成,系统中运行的各进程共享这些资源,必然出现争用而需要排队的现象,故可以用排队论来描述这类现象,用概率论来计算共享资源对运算速度的影响。

模型模拟方法 首先构造一个模拟模型来逼近目标系统。模拟模型包括系统模型和工作负载(环境)模型;工作负载模型又分为用户程序负载和系统程序负载。系统模型和工作负载模型是相互联系和相互影响的,一般采用面向对象技术和程序描述语言来描述。此外,还有计算机模拟程序包可供选用。模拟模型建立后,需要验证它的合理性、正确性;还需要设计模拟试验,对结果进行分析。

(2) 实际测试方法 机器研制完成后,可以用示波器、逻辑分析仪和测试程序等硬、软件工具,对计算机系统进行实际测试,这是最基本、最直接、最重要的系统运算速度评价手段:①硬件监视。在处理器设置(或外接)指令监视器,记录执行指令条数和运行时间。②软件监视。用相应测试程序监视执行指令条数和运行时间。③参考机比较。选定一个计算机系统做参考机,这个参考机的运算速度应当是已知的、公认的。再选定一组测试程序,这些程序应该是集中反映运算速度的核心程序、典型程序、吉布森混合比程序、基准程序和用户实用课题程序等。用上述程序分别在参考机和被测目标系统上运行,记录运行时间,则被测目标系统运行程序 i 的运算速度 R_i 为

$$R_i = \frac{t_{ir}}{t_i} \cdot R_r$$

式中 i ——第 i 个测试程序, $i=1, 2, \dots, n$;

t_{ir} ——第 i 个测试程序在参考机上运行时间;

t_i ——第 i 个测试程序在被测目标系统上运行时间;

R_r ——参考机平均运算速度, MIPS 或 MFLOPS。

可以取其算术平均值,则被测目标系统的平均运算速度 R 为

$$R = \frac{1}{n} \sum_{i=1}^n R_i$$

亦可取其几何平均值,则被测目标系统的运算速度为

$$R = \sqrt[n]{\prod_{i=1}^n R_i}$$

近年来,采用几何平均值的较多,参考机通常选用 DEC 公司的 VAX 11/780 和 Sun 公司的 sparcstation 10 Model 40,因为,这两种机器应用比较广泛,且公认 VAX 11/780 运算速度为 1 MIPS,而 sparcstation 10 Model 40 运算速度约为 VAX 11/780 运算速度的 40 倍。随着计算机技术的发展,基准程序和参考机选择都在变化,新的标准不断出现。

计算机系统运算速度是计算机系统主要性能指标。有关性能评价技术研究,使运算速度成为数量化的、能进行度量和评比的客观标准。所用基准程序已经回避了 I/O 的影响和操作系统开销,集中测试运算能力。当前,衡量通用处理器、高档微型计算机和工作站运算能力时,常用 SPEC 值来表示(参见 SPEC 基准程序),而衡量超级计算机时,通常用 LINPACK 基准程序来测试,主要测试浮点运算速度,测试结果用电子邮件在因特网上发送。1993 年开始,Jack. J. Dongarra 和 Hans. W. Meuer 对超级计算机进行了新的统计工作,每年两次公布全球超级计算机按性能前 500 名排行榜,基准程序是求解密集线性方程组,表中峰值运算速度 R_{peak} 用 MFLOPS 表示。从排行榜中,可以看出计算机发展的最高水平和发展趋势。

参考文献

1. 李三立,李亚民. RISC——单发射与多发射体系结构. 北京:清华大学出版社,1993

2. 罗晓沛,侯炳辉. 系统分析员教程. 北京:清华大学出版社,1992

(陈玉霜)

yunzuo guocheng

运作过程 (operation process) 用户和操作人员在使用用户的业务运作环境中为了使系统或软件产品投入运行所进行的一系列有关的活动。此过程包括对系统或软件产品的运作活动和用户运作时对他的支持活动。此过程的目的是在软件开发过程完成

后,将该系统从开发的环境移到用户的业务运作环境中运行;在运行时对用户的要求提供帮助和咨询;并对运作效果作出评价。此过程可为开发过程和维护过程提供有关的反馈信息,作为改进系统的依据。运作管理者可根据软件项目的总体要求按照**软件管理过程**的内容对运作过程进行管理。

运作过程一般包括以下活动:实施过程的准备;运作测试;系统转移到业务运作环境中;系统运作;对用户运作的支持;系统运作评价;用户运作评价等。

(1) 在开始准备时,要了解清楚软件开发过程的结果、准备用户的业务运作环境、制定运作过程的实施计划和运作评价标准。实施计划包括任务的分配、过渡计划(例如考虑人机并行运行)、操作培训计划、运作步骤等。

(2) 为了使系统转移到用户的业务运作环境中运行,操作者必须在业务运作环境中对系统进行运行测试。如果其测试结果能满足运行的基准要求,则可交付系统。

(3) 在系统向业务运作环境转移的过程中,必须进行以下工作:编制有关转移的文档;进行数据的转移;进行程序的转移;进行试运行;在试运行过程中进行跟踪;进行业务转移。当这些工作都完成后,进行转移的确认,并将转移结果的评价通知开发人员。

(4) 系统投入正式运行后,必须进行管理。运作管理者和操作者应当收集运行的数据,判别、记录 and 解决运作中发现问题,并改进运作环境。

(5) 操作者应当给予用户以运作上的支持。这包括:对用户进行操作培训和其他培训;应当建立接收、记录 and 解决用户请求的步骤;对用户的请求提供帮助和咨询服务,并对这些请求的响应进行记录和监控;必要时,操作者应当根据用户的请求向**软件维护过程**提供改进信息或修改请求,并由软件维护过程进行解决。

(6) 系统运作评价主要是指对系统的质量指标和其他方面评价。例如系统的响应分析、系统的运行效率、系统的安全性和保密性、系统故障分析、数据及媒体的管理以及人员管理、系统管理、运作时间管理等。

(7) 用户运作评价包括用户所要求的业务功能的实现程度、使用系统的容易程度、用户所设置的资源的运行和管理、系统运作效果以及用户对系统的改进要求等。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes — IEEE Std. 1074 ~ 1991
2. ISO /IEC 12207:1995 Information Technology — Software — Part 1: Software Life - Cycle Process

(刘光龙)

Z

zaoxing jishu

造型技术 (modeling technique) 采用计算机表示景物结构、形状和外观的方法与技术。它是计算机图形学的重要内容之一,也是计算机辅助设计的核心技术之一。根据所构造的对象不同,造型可分为规则形体造型和不规则形体造型。

对于形状规整的物体,如机器零件、建筑物、管道容器等,一般通过形体各部分的几何形状、空间位置以及各部分之间的连接关系来描述,达到建立其形状模型的目的。并且通过对模型的变换及操作,实现由简单模型构造复杂模型。这种方法称为**几何造型**。虽然几何造型技术已经得到广泛应用,但随着计算机辅助设计和制造一体化技术的发展,越来越暴露出几何造型技术的弱点。主要有:①几何造型只是反映了对对象的几何信息(几何坐标、拓扑关系等),而不能反映一个产品从设计到加工整个过程所需的全部信息(如公差、材料等),因而难于进行统一的考虑和处理。②几何造型只是从几何的角度描述一个对象,这将导致产品在设计 and 制造过程中信息处理的中断,人为干预多,使得计算机辅助设计及制造的一体化难于实现。在这样的背景下,就出现了基于特征的造型技术,称为**特征造型技术**。特征是产品设计与制造者感兴趣的对象。特征造型将特征作为产品描述的基本单元。在特征造型技术中,形状特征虽然往往是建立在几何造型的基础之上,但是用户直接是与高层次信息对话,避免了直接与几何、拓扑信息相联系,这就与设计者传统的设计方法更相一致。

基于物理的造型是几何造型技术的发展。在几何造型中,物体的形状和位置是直接由物体的几何数据和拓扑结构来表示的。而在复杂的动画画面或其他应用中,模型及模型间的关系相当复杂,不仅是静态的,而且是动态的。这时,靠人来定义物体的几何数据和拓扑关系非常繁杂,有时甚至是不可能的。这样,就出现了基于物理的造型技术。在这一技术中,物体的运动或变化规律由物理定律描述,根据物理规律产生运动或变化后,再通过几何模型产生相应的画面,在屏幕上显示出来。

在计算机艺术、计算机动画中往往涉及到大量的不规则形体,例如,山峦、树木、草丛、云、雾等。这些不规则形体的造型方法与规则形体不同,大多采用过程式模拟来实现。即用一个简单的模型及少量的易于控制的参数来表示一大类物体,不断改变参数,递归调用这一模型,就能逐步地产生出数据量很大的不规则物体,因而这一技术又称为**数据放大技术**。基于这一原理的不规则形体造型技术有**分数维造型**、**基于文法的造型**及**粒子系统**等。

参考文献

1. Mortenson M E. Geometric Modeling. New York: John Wiley & Sons, 1985
2. Foley J D, Dam A V et al. Computer Graphics: Principles and Practice. Addison-Wesley, 1990
3. 孙家广, 杨长贵. 计算机图形学(新版). 北京: 清华大学出版社, 1995 (杨长贵)

zengqiang xianshi

增强现实 (augmented reality, AR) 将计算机生成的虚拟对象和信息直接叠加在人可感知的真实世界之上的一种人机交互系统。增强现实系统具有以下3个特征:①真实世界和虚拟对象融为一体;②具有实时人机交互功能;③真实世界与虚拟对象是在三维空间上整合的。增强现实是虚拟现实的发展。在增强现实系统中用户不再与其所处的真实世界相隔离,而是处于真实世界与虚拟对象相融合的环境中。用户可以利用计算机生成的虚拟对象和信息来增强对真实世界的理解。这也是“增强现实”名称的来由。增强现实的“增强”特征首先体现在往所感知的真实世界中添加(叠加)虚拟对象和删除(隐蔽)真实世界中的对象。技术上,添加虚拟对象比删除真实对象较易实现。因此当前增强现实仅指往所感知的真实世界上添加虚拟对象。其次,“增强”特征应适用于所有视、听、触(力)、嗅等感觉通道的刺激信号。受技术限制,当前增强现实仅实现真实世界和虚拟对象视觉信号(图像)的融合。

构造增强现实系统的关键在于将真实世界与虚拟对象融为一体。这里有两类技术可以实现上述目

标,它们是光学技术和视频技术。应用光学技术实现融合的基本原理是通过一片半透明、半反射镜片实现融合,用户眼睛通过半透明镜片看到真实世界的同时,可以看到半反射镜片上的虚拟对象。应用视频技术实现融合的原理是通过摄像机将真实世界采样图像在图形处理器中叠加在虚拟对象图像上,再统一显示在屏幕上。根据显示器的不同,是采用透视式头盔显示器,还是采用台式图形显示器,又有两种不同的实现增强现实的方法。因此,增强现实系统的实现方法有 2 类 4 种:①基于光学的透视式头盔显示器实现方法;②基于光学的台式图形监视器实现方法;③基于视频的透视式头盔显示器实现方法;④基于视频的台式图形监视器实现方法。

增强现实的关键技术包括:①聚焦与对比度匹配 增强现实中的虚拟对象由计算机生成,所以它们不论远近总是清晰的;而摄像机摄取的真实世界图像可能部分清晰,部分模糊。另一方面,真实世界的动态对比度极大,而摄像机和图像监视器的动态对比度范围有限,因此在聚焦和对比度上都存在匹配困难问题。②系统的可移动性 增强现实的用户通常位于任务现场,且要求能在较大范围内移动,从而要求整个增强现实系统能随身携带,方便移动。③整合问题 增强现实系统对整合精度要求很高,且这种整合大多为动态整合,因此对实时性要求也很高。整合在时间和空间上的误差,严重时将使用户出现位置不适感。④传感器问题 为实现精确整合,要求能实时地、精确地、远程地跟踪用户头部位置和朝向,目前的传感器技术离上述要求尚有较大差距。

增强现实的应用原理在于通过将虚拟对象叠加在真实对象之上提示用户注意这些真实对象,或补充新的实际并不存在的虚拟对象,或通过相应解释性文字等信息,帮助用户理解真实对象。增强现实系统已在医学可视化、维护与修理指导、解释与提示、机器人路径规划、导游、娱乐、建筑工程规划、飞机导航和攻击瞄准等领域获得应用。

参考文献

1. Azuma Ronald T. A Survey of Augmented Reality. Presence, August 1997, 6(4): 55~385
2. 石教英主编. 虚拟现实基础及实用算法. 北京:科学出版社,2002 (石教英)

zhan zidongji

栈自动机 (stack automaton) 一种受限型的

图灵机(图 1)。它含有:①有限控制器;②具有端点标志的输入带和双向输入读头,输入带只读不写;③一存储带或栈,它可以用作下推存储,读头不能印或抹(即印空白),除非它的右面全是空单元。但读头可以按只读(不写)方式在带的非空部分内到处游动。

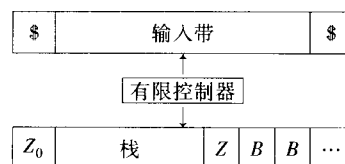


图 1 栈自动机

栈自动机类似于下推自动机。不同之处在于它可以在输入带上左右移动读得输入符号,也可以在栈的任何位置上读得符号,但改写符号只能在栈顶。栈自动机比下推自动机识别语言的能力强,但仍大大弱于图灵机。栈自动机一般可看成为一种非确定型的受限图灵机,可将其修改为确定型非抹除的(即任何栈符号都不被抹除)。已经证明:如果语言 L 可被一确定的非抹除的栈自动机接受,则它可被一确定的带复杂度为 $n \log n$ 的图灵机接受。栈自动机及其许多变种在接受语言方面的性质已被广泛研究。(李祥)

zhengui yizhijie qijian

锗硅异质结器件 (heterojunction devices of SiGe/Si) 利用 $\text{Si}_{1-x}\text{Ge}_x$ 和 Si 所形成的异质结构做成的半导体集成电路。与半导体经掺杂后形成的 PN 结(同质结)不同,异质结是由两种不同的半导体之间形成的结。由于形成异质结的两种材料的禁带宽度、介电常数、折射率和吸收系数等物理参数不同,使异质结具有许多同质结所没有的优异特性,可以获得更高的器件放大倍数和响应速度。因此,利用异质结制作的器件得到极大重视。现在已成功地研制出 $\text{GaAs}/\text{Al}_x\text{Ga}_{1-x}\text{As}$ 双异质结激光器、长波长 $\text{InP}/\text{InGaPAs}$ 激光器、 $\text{GaAs}/\text{Al}_x\text{Ga}_{1-x}\text{As}$ 和 $\text{InP}/\text{In}_x\text{Ga}_{1-x}\text{As}$ 做成的异质结双极晶体管(HBT)和高电子迁移率晶体管(HEMT)等性能优越的异质结器件。

$\text{Si}_{1-x}\text{Ge}_x$ 异质结构(包括超晶格和多量子阱)是 20 世纪 90 年代发展起来的一种新型光电子和微电子材料。它具有许多优异的特性。应变 SiGe 的禁带宽度比 Si 小,可在 1.1 eV 和 0.7 eV 之间调节,

SiGe 应变超晶格可以是直接带间跃迁。这些能带的特点使 SiGe/Si 异质结构不仅可以用来制造高速、高增益电子器件并加以集成,而且可以用来制造光电子器件并加以集成。SiGe 器件工艺与硅微电子器件工艺可以兼容,因此 SiGe 器件能够与硅集成电路或硅基集成电路集成在一起,而且可以在硅集成中开拓硅光集成,从而使微电子器件和光电子器件基质材料的不统一得到解决。SiGe 比 GaAs 等 III-V 族化合物半导体材料便宜,使器件成本大大降低。由于 SiGe/Si 优越的材料和器件特性,许多国家已投入相当的人力和物力,对材料生长和器件开发等进行了广泛和深入的研究,不断地研制出具有高水平的 SiGe 器件。

异质结双极晶体管 (HBT) 是以 SiGe 作为基区的异质结双极晶体管。在 Si 器件和电路中,频率最高、速度最快的属双极型晶体管 (BJT) 及由它组成的双极电路。SiGe HBT 与 Si BJT 相比,具有传输时间短、截止频率高、电流增益大、低温特性和噪声特性好等优点。器件的频率特性和电路的延迟时间性能已达到 GaAs 器件的水平。当 BJT 的发射区用宽禁带材料 Si,基区采用窄禁带材料 SiGe 制备时,这种异质结可以提高发射区向基区的载流子注入和抑制基区向发射区的载流子注入,大大提高电流放大系数 β , 减少发射结电容,有利于提高截止频率 f_T 。采用这种异质结构,还可以提高基区的掺杂浓度,减少基区电阻,使基区可以做得更薄,大大提高器件的工作频率。如果基区禁带宽度设计为缓变的,不仅可以缩短载流子在基区的渡越时间,大大提高 f_T ,而且可以使器件在低温下也能高速工作。已研制成功 NPN SiGe/Si HBT 和 PNP SiGe/Si HBT 等器件 SiGe HBT 的 f_T 最高值已达 452 GHz,也是硅基晶体管和双极型晶体管的最高值。

调制掺杂场效应晶体管 (MODFET) 是利用 Si 和 SiGe 形成的调制掺杂异质结构做成的场效应晶体管 (FET)。根据禁带宽度差和带边移位的特点,MODFET 可以分为两种结构,即电子量子阱和空穴量子阱的 MODFET,它们分别称为 N 沟 MODFET 和 P 沟 MODFET。N 沟 MODFET 是在 Si 衬底上生长一层 SiGe 缓冲层,再外延生长一层不掺杂的 Si 层和 N 型掺杂的 SiGe 层,即所谓调制掺杂。由于 Si 膜含有张应力,导致带下降形成电子势阱和二维电子气传输沟道。沟道层 Si 和电子供给层 SiGe 是分开的,可以消除场效应晶体管中通常存在的杂质散射,使电子迁移率大大提高。所以这种器件又称为高电子

迁移率的晶体管。P 沟 MODFET 是在 P⁻Si 衬底上生长一层不掺杂的 SiGe 应变层和 P⁺Si,形成 Si/SiGe/Si P 型调制掺杂的量子阱结构。它可分为增强型和耗尽型的 P-MODFET。

双极反型沟道场效应晶体管 (BICFET) 一种利用反型沟道而不是电中性基区来调制集电极电流的场效应异质结双极晶体管,其主要特点是没有基区。它是借助场效应机制产生一反型沟道层,其作用相当于普通双极晶体管的基区。BICFET 有三个端:金属发射极、源极和集电极。金属发射极与半绝缘层 (宽禁带半导体) 形成欧姆接触,源极、半绝缘层和半导体耗尽区界面处形成反型层接触,集电极由半导体材料做成。其工作原理是利用反型沟道中电荷的偏置作用控制多数载流子从半绝缘区到集电极的流动。BICFET 是一种新型的器件,它有很高的电流增益 (约为 10^5),很高的工作电流 (10^6 A/cm²) 和跨导 (4×10^7 S/cm²),以及低的输入电容。这种器件的结构克服了像 MOS 器件或结型双极晶体管中由于穿通而带来的比例限制问题。

BICFET 可用 Si、III-V 族化合物半导体和 SiGe 等材料制备。但 III-V 族化合物异质结能带结构不适宜制备 P 沟 BICFET,而 SiGe/Si 应变异质结是一种较理想的材料。P 沟 BICFET 的研制已取得初步结果,室温下增益 β 可达 1 720, 77 K 下 β 可达 3 000。器件结构和工艺有待进一步改进,但可以预见这种器件有望成为一种新的高速、高灵敏度和大功率的器件,并用于超大规模集成电路 (VLSI) 中。

共振隧穿器件 (RTD) 利用异质结双势垒的载流子共振隧穿 (RT) 效应制成的器件。RT 结构可以由单阱双势垒组成,也可以由双阱三势垒组成或短周期超晶格组成。RT 结构除可以制备成二极管外,还可以制备成双极晶体管,即 RTBT。在 RTBT 中,RT 结构可以设置在发射区内,也可以设置在基区内。它是一种很有前途的功能器件,电路方面应用很广,可以作模拟量和数字量的转换电路,倍频电路,毫米、亚毫米波放大器和多值存储电路等。主要特点是能简化电路的复杂程度。自从 SiGe/Si 应变异质结出现后,人们已采用这种材料来研制共振隧穿器件,已制备出空穴 RTD 和电子 RTD。

在 VLSI 芯片中含有数千万个以上的元件,随集成度进一步提高,其内部连线将达到几何尺寸的极限 0.1 μ m。再进一步的发展中,预计 RTBT 和量子耦合器件将在电路功能方面起重要作用,而且有可能运用隧穿作用在器件之间连线。另外,应用 RT

晶体管的多状态特性将可以设计出运用多值逻辑电路的新的计算机结构。

1.3 ~ 1.5 μm 波长的 SiGe/Si 光探测器 SiGe 合金材料禁带宽度在 1.12 ~ 0.7 eV 之间可以调节,对应的波长为 1.0 ~ 1.5 μm 。SiGe/Si 应变层由于应力存在,其禁带宽度可以降得更低,使 SiGe 材料的光吸收波长提高到适于光纤通信用的 1.3 ~ 1.5 μm 的波长范围。应用 SiGe/Si 应变超晶格材料作光吸收层,已研制出可在 1.3 ~ 1.5 μm 范围工作的 P 型-本征耗尽层-N 型半导体 (PIN) 光探测器,雪崩型的光电探测器 (APD) 等高光电增益的器件。但 SiGe/Si 应变超晶格 PIN 和 APD 探测器研究成果不如 SiGe 基区 HBT 那么突出和显著,主要原因是 SiGe/Si 应变材料对 1.3 μm 、1.5 μm 波长的光吸收系数较小,而探测器暗电流较大。一旦有效地解决这两个问题, SiGe/Si 探测器的性能定会进一步提高,并在长波长光纤通信中发挥作用。

长波长红外探测器 (LWIRD) 波长在 3 ~ 5 μm , 8 ~ 12 μm 甚至更长范围的红外探测器。SiGe/Si 异质结制备的 LWIRD 器件是采用高掺杂简并的 P⁺ SiGe 作发射区, P⁺ Si 衬底作收集区。红外波段的光子在 P⁺ SiGe 区内被吸收后在价带内发射出空穴,空穴在外加电场作用下,越过 SiGe/Si 之间的势垒被 Si 衬底收集形成光电流,从而实现对红外光子的探测。SiGe/Si 异质结的势垒高度可以由 SiGe 层中 Ge 含量调节,当 Ge 含量在 0.1 ~ 0.4 之间时对应的截止波长可在 5 ~ 22 μm 范围变化。这是其他探测器所不具备的优点。SiGe/Si LWIRD 可以与有关电路单片集成,可以制作大规模集成的远红外探测器阵列。目前主要研究方向是进一步降低暗电流和提高量子效率。

SiGe/Si 数字型光开关 (DOES) 一种双稳态的 SiGe/Si 异质结构器件。这一器件的特点是具有微分负阻区分开的高阻抗的关态和低阻抗的开态。当施加电压或电流超过高阻抗态对应的电压 U_s 或电流 I_s , 或通过入射光照时,器件从关态转换到开态;当电流或电压减少到低于 I_s 或 U_s , 或无光照时,器件从开态转换到关态。

半导体异质结构、量子阱、超晶格材料的出现,使半导体器件的设计和制作由“杂质工程”发展到“能带工程”。基于 SiGe 的能带工程,原则上 SiGe 可以代替 Si 发展各种微电子器件。SiGe/Si 器件在高速、大功率和低噪声等方面都有突出的优点。基于 SiGe HBT,可以发展许多性能优异的微波器件和

高速电路及微波集成,甚至发展 SiGe HBT 集成电路的超高速计算机和超快速微电子器件系统。利用 SiGe RTD 可以在 Si 基 VLSI 中发展超高频的数字逻辑电路。应用 RTD 的多状态特性可能设计出运用多值逻辑状态的新的计算机结构。通过改变 SiGe/Si 异质结构、量子阱、超晶格的组分,应变层厚度及掺杂浓度等,还可以发展各种新型的 Si 基发光器件与集成电路,使探测器性能得到较大提高。Si 微电子工艺可使 SiGe 材料用于发展光通信、光计算的硅锗单片集成电路,并能够实现大规模集成的远红外探测器阵列和焦平面的大规模集成。

参考文献

1. Rieh J-S, Jagannathan B, Chen H et al. SiGe HBTs with cut-off Frequency of 350 GHz. IEDM'02 Dig., 2002:771 ~ 774
2. Khaled Ismail, Rishton S, Chu J O et al. High-Performance Si/SiGe n-Type Modulation-Doped Transistors. IEEE Electron Device Letters, 1993, 14(7): 348 ~ 350

(陈维德)

zhenshiguan tuxing shengcheng

真实感图形生成 (realistic graphics generation) 使三维空间的物体生成具有色彩、纹理、阴影、层次等真实感图形的过程和技术。又称真实感图像综合。其目的是对于空间的各种物体和自然景物,利用计算机图形生成技术产生恰如拍照片一样的真实感效果。

为了产生图形的真实感,一般需要解决以下几方面的图形综合技术问题:

(1) 在图形中消除在特定观察点看不见的物体或部分物体,从而产生空间物体的层次感(参见消隐技术);

(2) 在物体表面生成各种各样的纹理,以增强物体的质感(参见纹理生成);

(3) 尽可能精确地模拟光源照射的物理效果,使空间物体具有像拍照片一样的光照效果和明暗层次(参见光照模型、光线跟踪技术、辐射度技术);

(4) 模拟透明物体的效果;

(5) 在显示设备有限的离散精度范围内,尽量保持图形具有自然的光影过渡和连续性(参见图形反走样技术)。

使计算机生成的图形具有逼真的感觉是许多领域所追求的目标。该技术在计算机动画(如影视、广告、计算机艺术等)、计算机仿真、计算机辅助设

计、计算机辅助制造、科学计算可视化以及虚拟现实等众多领域已得到广泛应用。真实感图形生成技术可模拟生成更为复杂的自然场景和具有更高的逼真度,特别是在影视、虚拟现实等领域得到了进一步的发展和应用。

参考文献

1. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2001
2. 孙家广,杨长贵等. 计算机图形学(新版). 北京:清华大学出版社,1995
3. 唐泽圣,周嘉玉,李新友. 计算机图形学基础. 北京:清华大学出版社,1995 (吴思华)

zhenlie chuliji

阵列处理机(array processor) 在同一控制器控制下,按同步方式工作的处理器阵列。阵列处理机是一种典型的单指令流多数据流(SIMD)计算机。

在阵列处理机中,指令由阵列控制器译码,阵列中所有的处理器都执行同样的操作,只是处理的数据不同。

1958年,F. H. Unger 为求解空间问题设想了一个主控制器控制的二维处理单元阵列结构;1962年,D. L. Slotnick 等提出的 Solomon 计算机,进一步肯定了 SIMD 概念。第一台大型的 SIMD 阵列处理机是 60 年代末由伊利诺依大学设计、1972 年由 Burroughs 公司生产的 Illiac IV (64 个处理单元)。此后,Burroughs 公司又设计了性能更高的 BSP 阵列处理机。为了满足图像处理等领域的应用需求,70 年代末和 80 年代初出现了一些由大量位片处理器构成的阵列处理机系统,如英国的 CLIP 和 DAP,美国的 MPP 等。1985 年,Thinking Machine 公司的 CM-1 使用了 65 536 个位片处理单元。进入 90 年代后,MIMD 型 MPP 系统开始与 SIMD 阵列处理机竞争市场,但仍有一些阵列处理机产品问世,如 MasPar MP-1 (16 384 个处理单元)和 MP-2 (65 536 个处理单元)。另外,一些半导体厂家也开始推出适合图像或数字信号处理等特定应用的单片阵列处理器。

阵列处理机通常不是一台独立的计算机,它需要一台宿主机作为用户界面,并将程序和数据装入阵列处理机的控制器中的存储器。但新一代的阵列处理机,如 MP-1 等,已将宿主机功能集成在系统中。

在阵列处理机中,除了阵列控制器和处理单元阵列外,还有一个标量处理器。阵列处理机的指令一般分为标量指令和向量指令两类,阵列控制器完成指令译码后,将标量指令交给标量处理器执行,而将向量指令广播给所有处理单元执行。另外,通过条件指令可以设置屏蔽,以临时禁止某些处理单元参加运算。

阵列处理机有分布式存储器和共享存储器两种类型,前者是主流。分布存储的阵列处理机中,处理单元包含运算部件和局部存储器两部分,各处理单元间通过点对点的交换网络(如二维网格、超立方体等)相连。共享存储阵列处理机的典型代表是 BSP,其处理单元与存储器模块通过一个对准网络相连。为了减少存储体的冲突,BSP 使用了 17 个存储模块(即素数个存储模块)。

参考文献

- 黄铠,Briggs F A 著. 计算机结构与并行处理. 金兰等译. 北京:科学出版社,1990 (唐志敏)

zhengze biaodashi

正则表达式(regular expression) 对有限自动机所接受的语言或时序开关电路的行为的形式描述。它告诉我们,如何使用正则运算从原子语言构造一般语言。原子语言为空语言 φ 和单元集 $\{a\}$, 其中 a 是预先指定的字母。正则运算为并、连接和连接闭包。“并”是通常集合论中的“并”运算。两个语言 X, Y 之间的连接 XY 由形如 xy 的字组成,其中 $x \in X, y \in Y$ 。语言 X 的连接闭包 X^* 由空字和所有形如 $x_1 \cdots x_n$ 的字组成,其中 $n \geq 1, x_i \in X$ 。例如,以任意方式连接 ab 和 b 得正则表达式 $(ab \cup b)^*$, 所得语言 X 为字母表 $\{a, b\}$ 上由空字和以 b 结尾且不含子字 aa 的所有字组成的语言。

正则表达式的形式定义: 假设 V 和 $V_1 = \{\varphi, \cup, *, (,)\}$ 是不相交的字母表。字母表 $V \cup V_1$ 上的字 α 称为 V 上的一正则表达式,恰当 α 是下述情形之一:

- (1) α 是 V 的字母,或字母 φ ,
- (2) α 形如 $(\beta \cup \gamma), (\beta \gamma), \beta^*$ 之一,其中 β, γ 是 V 上正则表达式。

按下述约定, V 上每个正则表达式 α 表示 V 上的一语言 $|\alpha|$:

- (1) φ 表示的语言是空语言,
- (2) $a \in V$ 表示的语言由字 a 组成,
- (3) 对 V 上正则表达式 $\beta, \gamma, |(\beta \cup \gamma)| = |\beta| \cup |\gamma|$

$$|\gamma|, |(\beta\gamma)| = |\beta| |\gamma|, |\beta^*| = |\beta|^*.$$

看上去很不相同的正则表达式可标记相同语言。例如,如下正则表达式

$(a \cup ab \cup ba)^*, (ba \cup a^* ab)^* a^*, a^* (ab \cup ba^* a)^*$ 表示相同语言。

有限自动机和时序开关电路所对应的正则表达式化简后,相应行为变得很好理解。在正则表达式与有限自动机之间,存在许多相互转化的算法。

作为正则表达式的应用实例,许多操作系统、屏幕编辑器、字处理程序已扩展了它们的字符串搜寻能力,使得除寻找特殊串外,还可用来匹配一特定正则表达式的串。这种搜寻工具的最好例子是 Unix 的 grep 命令,它表示“获得正则表达式”。

参考文献

Hopcroft J E, Ullman J D 著. 自动机理论、语言和计算导引. 徐美瑞译,洪加威校. 北京: 科学出版社, 1986 (陈火旺 贵可荣)

zhengze wenfa

正则文法 (regular grammar) 左线性或右线性文法。

一个左线性文法可用四元组 $G = (V, \Sigma, P, S)$ 表示,其中 V 是变元的有限集合, Σ 是终结符的有限集合, $S \in V$, 称为开始符号, $w \in \Sigma^*$ (即 w 为有限个终结符连接成的串或字,可能为空串或空字 ε)。 $A, B \in V$ 时, P 是由形为 $A \rightarrow w$ 和 $A \rightarrow wB (A \rightarrow Bw)$ 产生式组成的有限集。右线性文法与左线性文法是等价的,即可生成同样的语言(字集合)类。

正则文法来源于 20 世纪 50 年代中期 N. Chomsky 对自然语言的研究,是乔姆斯基短语结构文法分层里的 3 型文法。正则文法类是上下文无关(2 型)文法类的真子类,已应用于计算机程序语言编译器的设计、词法分析(文本处理中描述触发过程动作的文本模式、文件类型和扫描器、文本工具的标准基础)、开关电路设计、句法模式识别等,是计算机和信息科学、工程、物理、化学、生物、医学、应用数学不可忽视的论题。

正则文法的结构与复杂性测度由变元、产生式的个数及文法有向图的高度、每一层的结点数来确定。 $S \xrightarrow{+}_c w$ 表示有限次使用 P 中产生式可派生出字 w , 正则文法 G 可作为生成器产生和描述正则语言 $L(G) = \{w \in \Sigma^* \mid S \xrightarrow{+}_c w\}$ 。例 1. $G = (\{S, A, B\}, \{0, 1\}, P, S), P = \{S \rightarrow 0A10, A \rightarrow 1B, B \rightarrow 0A10\}, G$

是一个正则(右线性)文法, $L(G)$ 中含字 $0, (S \rightarrow 0), 01010 (S \rightarrow 0A \rightarrow 01B \rightarrow 010A \rightarrow 0101B \rightarrow 01010)$ 。正则语言也称为正则集,可以用正则表达式表示。对任一正则表达式,可以构造出带 ε 动作的非确定有限自动机(NFA)在线性时间内来接受它,也可构造出不带 ε 动作的确定有限自动机(DFA)在平方时间内来接受它,正则文法生成的语言也可由双向确定有限自动机(2DFA)来接受, NFA, DFA, 2DFA 是等价的,即所接受的语言类是相同的。

正则表达式 递归地定义如后,设 Σ 为有限集, ① \emptyset, ε 和 $a (\forall a \in \Sigma)$ 是 Σ 上的正则表达式,它们分别表示空集、空字集 $\{\varepsilon\}$ 和集合 $\{a\}$; ②若 α 和 β 是 Σ 上的正则表达式,则 $\alpha \cup \beta, \alpha \cdot \beta = \alpha\beta$ 和 α^* 也是 Σ 上的正则表达式,它们分别表示字集 $\{\alpha\}, \{\beta\}, \{\alpha\} \cup \{\beta\}, \{\alpha\} \cdot \{\beta\}$ 和 $\{\alpha\}^*$, (运算符 \cup, \cdot 和 $*$ 分别表示并、连接和星(乘幂闭包) $\{\alpha\}^* = \{\bigcup_{i=0}^{\infty} \alpha^i\}$), 优先顺序为 $*, \cdot, \cup$; ③只有有限次使用①和②确定的表达式才是 Σ 上的正则表达式,只有 Σ 上的正则表达式所表示的字集才是 Σ 上的正则集。如例 1 中正则文法生成的字集,其正则表达式为 $0(10)^*$ 。为了简化正则表达式,常用下列等式: ① $\alpha \cup \alpha = \alpha$ (幂等律); ② $\alpha \cup \beta = \beta \cup \alpha$ (交换律); ③ $(\alpha \cup \beta) \cup \gamma = \alpha \cup (\beta \cup \gamma)$ (结合律); ④ $\alpha \cup \emptyset = \alpha, \alpha \emptyset = \emptyset \alpha = \emptyset, \alpha \varepsilon = \varepsilon \alpha = \alpha$ (零一律); ⑤ $(\alpha\beta)\gamma = \alpha(\beta\gamma)$ (结合律); ⑥ $(\alpha \cup \beta)\gamma = \alpha\gamma \cup \beta\gamma$ (分配律); ⑦ $\varepsilon \cup \alpha^* = \alpha^*$; ⑧ $(\varepsilon \cup \alpha)^* = \alpha^*$ 。用①至④可将 α 变为 β 时,称 α 与 β 相似。正则表达式的导式类似于微积分学中的导式,可用于正则语言的商运算, α 关于 $x \in \Sigma^*$ 的导式 $D_x \alpha$ 递归定义如后: ① $D_\varepsilon \alpha = \alpha$; ② $\forall x \in \Sigma, D_x \emptyset = \emptyset, D_x \varepsilon = \emptyset, D_x a = \begin{cases} \emptyset, & x \neq a \\ \varepsilon, & x = a \end{cases} (\forall a \in \Sigma)$; ③ $\forall x \in \Sigma, D_x (\alpha \cup \beta) = D_x \alpha \cup D_x \beta, D_x (\alpha\beta) = (D_x \alpha)\beta \cup \alpha(D_x \beta), D_x (\alpha^*) = (D_x \alpha)\alpha^*$; ④ $\forall x = x_1 \cdots x_n \in \Sigma^*, D_x \alpha = D_{x_n} (\cdots (D_{x_2} (D_{x_1} \alpha)) \cdots)$ 。每一正则表达式只有有限个不相似的导式。表 1 给出了正则表达式复杂性的两种测度。运用正则表达式方程 $X_i = \alpha_{i_0} + \alpha_{i_1} X_1 + \cdots + \alpha_{i_n} X_n$ 来处理语言有其方便之处,因为这种方程中 $\Delta = \{X_1, \cdots, X_n\}$ (未知量的集合)与 Σ 之交为 \emptyset, α_{ij} 为 Σ 上正则表达式,当 α_{ij} 为 \emptyset, ε 时分别相当于普通线性方程组之系数 0, 1, 可以按线性方程组的高斯消去法求解,当然,这里的解是一个集合,即解不是惟一的,但该算法能够正确地确定一个极小不动点作解。

表 1

测 度 \ 正则表达式	\emptyset	ε	a	$\alpha \cup \beta$	$\alpha\beta$	α^*
H (* 在表达式中嵌套的层数——星高)	0	0	0	$\max\{H(\alpha), H(\beta)\}$	$\max\{H(\alpha), H(\beta)\}$	$H(\alpha) + 1$
N (表达式中出现的符号个数)	0	0	1	$N(\alpha) + N(\beta) - \{\text{重复出现于 } \alpha, \beta \text{ 中符号个数}\}$	$N(\alpha) + N(\beta) - \{\text{重复出现于 } \alpha, \beta \text{ 中符号个数}\}$	$N(\alpha)$

正则文法的性质 由所生成的正则语言来体现。如果 R 为正则语言, 则存在一个常数 n , 使得 R 中所有字长不小于 n 的字 w 都可写成 xyz 的形式 ($y \neq \varepsilon$ 且 $|xy| \leq n$) 并且对所有非负整数 i 必有 $xy^i z \in R$, 此为泵引理。它是证明某些语言不正则的有力工具, 且有助于建立算法来判断一个给定的正则文法所生成的语言是有限的还是无限的。判断某些语言是否正则还可以利用对语言运算是否封闭来决定。已知正则语言类对布尔运算(并、交、补)、连接、* (克林尼闭包)、左右商、替换、同态、逆同态、INIT(求前缀)、FIN(求后缀)、MIN、MAX、CYCLE、Reversal 等封闭。又当 $p(x)$ 为非负整系数多项式, R 为正则语言时, $L_1 = \{w \mid \text{对某个使 } |y| = p(|w|) \text{ 的 } y \text{ 有 } wy \in R\}$, $L_2 = \{w \mid \text{对某个使 } |y| = |w| \text{ 的 } y \text{ 有 } wy \in R\}$ 也是正则语言。当 R, R_1 和 R_2 是正则语言时下列问题都是可判定的: $w \in R? R = \emptyset? R = \Sigma^*? R_1 = R_2? R_1 \subseteq R_2? R_1 \cap R_2 = \emptyset?$

正则文法的推断 模式识别中研究模式类不可缺少的工具之一。即根据具体问题的特性, 从其样本集 $R^+ (\subseteq L(G))$ 及其否定 R^- (R^+ 关于 $L(G)$ 的补集), 用算法来找出正则文法 G 就是正则文法的推断。确定的 G 不仅能生成 R^+ , 而且可以生成有关模式更多的样本、事实和问题的其他解答。正则文法是最容易实现推断算法公式化的文法, 而且必定可以最小化。正则文法的推断方法有三: ①交互作用, 即借助教师(操作员)来判定字的合法性及其性质; ②推广规范文法 $G_1 (L(G_1) = R^+)$ 为 G ; ③从接受 R^+ 的有限自动机来得到 G 。

并行正则语言 适应大规模并行计算的需要。仍使用语言运算来研究, 除 \cup 、 \cdot 和 $*$ 三种基本运算以外, 还引入了四种运算: “ \parallel ”(交织), $\forall a \in \Sigma, a \parallel \varepsilon = \varepsilon \parallel a = \{a\}, \forall a, b \in \Sigma, s, t \in \Sigma^*, as \parallel bt = a(s \parallel bt) \cup (as \parallel t)b, \forall A, B \subseteq \Sigma^*, A \parallel B = \{x \mid \text{有 } s \in A, t \in B, \text{使 } x = s \parallel t\}$ (注: 含 \parallel 的表达式可以归约为不含 \parallel 的正则表达式); “[]”(同步合成),

$\forall A \subseteq \Sigma_A^*, B \subseteq \Sigma_B^*, A[] B = \{x \mid x / \Sigma_A \in A, x / \Sigma_B \in B\}$, (x / Σ 表示字 x 在 Σ 里的符号串); “ σ ”(再命名), $\sigma(R) = \{\sigma(x) \mid x \in R\}$, 如果 $\sigma(\varepsilon) = \varepsilon$ 表示 σ 可隐蔽, 如果 $\sigma(a) = \sigma(b)$ 表示 σ 可以部分观察, 如果 $\sigma(a) = a$ 表示 σ 为相似过程或遗传; “ α ”(Alpha 闭包), $A^\alpha = \bigcup_{i=0}^{\infty} A^{(\alpha)^i}$ 。在上述七种运算下封闭的正则表达式与佩特里网语言相联系, 从而用于研究并行计算。

ω 正则文法 正则文法产生式中允许 ω 为无限串则生成 ω 正则语言, 也得到不少研究成果, 如豪斯朵夫-库拉托夫斯基分层, 莫勒自动机分层。

正则文法与抽象代数 正则集与半群(服从结合律的二元运算的非空集)、么半群(有么元的半群)、态射和直积等抽象代数内容相联系已形成学科性课题, 已经证明: ① $R \subseteq \Sigma^*, \Sigma^*$ 的右同余(等价)关系 E_R 规定为: $\forall x, y \in \Sigma^*, x E_R y \Leftrightarrow \forall z \in \Sigma^*$ 有 xz 和 yz 都在 R 中或都不在 R 中。 R 为正则语言 $\Leftrightarrow \Sigma^*$ 关于 E_R 的等价类类数有限。② Σ^* 对连接运算构成一个么半群, R 为正则语言 \Leftrightarrow 存在有限么半群 N 和同态映射 $\varphi: \Sigma^* \rightarrow N$ 及 $T \subseteq N$ 使得 $R = T\varphi^{-1}$ 。③令 $PF(Q)$ 为 Q 到 Q 的全体偏函数集, $PF(Q)$ 对函数合成运算构成一个么半群, 同态映射 $\varphi: \Sigma^* \rightarrow PF(Q) (x = x_1 \cdots x_n \in \Sigma^*, \varphi(x) = \varphi_x \in PF(Q))$ 下 R 的象 M_R 称为 R 的语法么半群。当 Q 为接受语言 R 的自动机 M 的状态集时, $\varphi_x(q) = \varphi_m(\cdots(\varphi_{x_1}(q)) \cdots), q \in Q, \varphi_{x_i}(q) = \delta(q, x_i), \delta$ 是 M 的转移函数。 R 为正则集 $\Leftrightarrow M_R$ 为有限集。④每一正则语言 R 都存在同态映射 h_1, h_2, h_3, h_4 使 $R = h_4 h_3^{-1} h_2 h_1^{-1} (1^* 0)$ 。⑤塞缪尔·爱伦堡定理: 正则语言的星流形(指若干个正则语言形成的族在布尔运算、派生、逆同态下封闭)与有限么半群流形(指若干个有限么半群形成的族在态射象、子么半群、有限直积下封闭)之间存在一一对应。

Büchi 建立了正则文法、有限自动机与逻辑的关系。Higman, Kundu 分别对识别正则语言的子串、

词、字典、前缀、后缀的有限自动机的状态数的上下界进行了研究。

参考文献

1. 陈火旺, 钱家骅, 孙永强. 程序语言编译原理 (第二版). 北京: 国防工业出版社, 1984
2. Hopcroft J E, Ullman J D. Introduction to Automata Theory, Languages, and Computation. Reading, Mass.: Addison-Wesley, 1979
3. Gonzalez R C, Thomason M G 著. 句法模式识别. 濮群, 徐凤家, 徐光祐译. 北京: 清华大学出版社, 1984 (张一立)

zhengzhongji jiaohuan

帧中继交换 (frame relay switching) 在链路层通过逻辑链路的复用和转接以实现以帧为单位的交换。这是在数据链路层实现网络资源统计复用的一种快速分组交换技术。这里的网络资源指的是通信网络的资源, 它包括线路的传输能力和交换机的交换能力, 而统计复用实质上就是按需分配的意思。帧中继的出现, 一方面反映了用户应用要求的提高, 即要求通信网能提供高的传输速率、快速响应时间和突发的信息传输等, 另一方面, 由于光纤的普及, 通信线路传输速率和质量的提高, 有可能简化交换网的通信协议以提高通信效率。帧中继技术是对 X.25 分组交换技术的一种改进 (参见分组交换)。为了说明帧中继工作原理, 图 1 给出了它与 X.25 的比较。

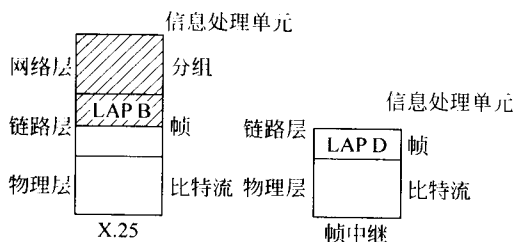


图 1 帧中继与 X.25 比较

X.25 网使用开放系统互连基准 (参考) 模型的下 3 层协议。由于 X.25 分组交换网从源点到终点的每一步, 每个交换机都要进行大量的处理, 从而导致延迟大且有效传输速率低。

帧中继只有物理层和数据链路层两层协议。物理传输信道采用高质量的光纤, 它的误码率低、传输速率高。数据链路层执行的只相当 X.25 的数据链

路层的一部分功能。它保存了 X.25 数据链路层的高级数据链路控制规程的帧格式, 但不是采用 X.25 的 LAP B 规程, 而是采用了 ISDN 的 D 通道的规程 LAP D。这样, 帧中继能够在链路层实现逻辑链路的复用和转接, 而 X.25 则是在网络层实现复用和转接, 因此, 帧中继可以不用网络层就能实现帧级的复用和传送。

图 2 给出了 X.25 LAP B 帧格式与帧中继的帧格式。

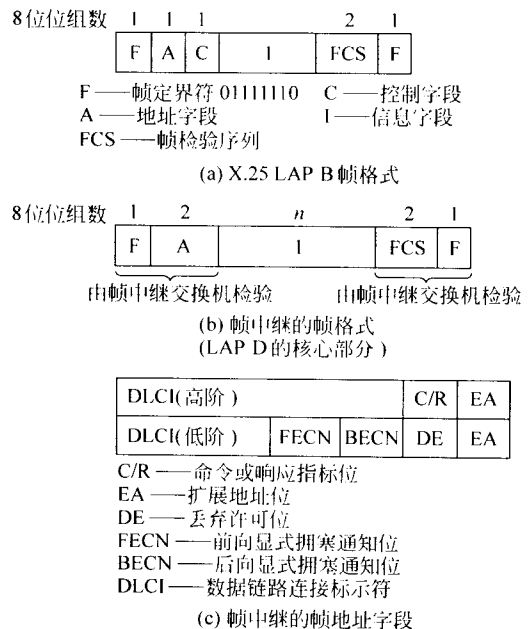


图 2 X.25 LAP B 帧格式和帧中继的帧格式

帧中继采用 LAP D 规程。每个帧含有数据链路连接标识符 (DLCI), 用以标识虚电路, 实现基于永久虚电路 (PVC) 的连接。由前向显式拥塞通知 (FECN)、后向显式拥塞通知 (BECN) 和丢弃许可 (DE) 位进行流量控制和检验。帧中继交换机只在超载情况下向 DTE 发送 FECN 或 BECN 信号, 要求减少数据量。当已发生超载时, 网络就丢弃帧, 而由终点发现时请求重发。帧首定界符, DLCI, 帧检验序列 (FCS) 和帧尾定界符在中继交换机进行检验, 但不计数, 不要求重发, 发现有错, 即行丢弃, 只在终点和源点设备间负责检错重发, 从而简化了帧中继交换机的处理过程。图 3 给出了帧中继与 X.25 的差错处理的比较。

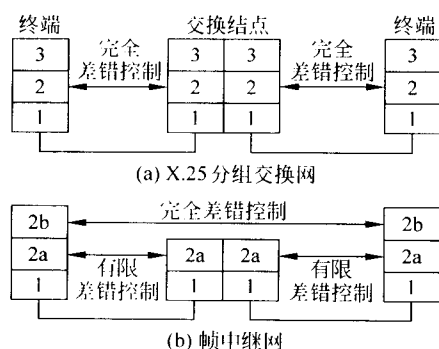


图3 帧中继与 X.25 的差错处理的比较

帧的用户信息(1)字段,可以是 X.25 分组或其他格式,如 LAN 的帧、SNA 的帧等,由于帧中继无 OSI 第三层功能,用户信息在网内透明传输,不作任何处理。因此,帧中继网很容易实现与其他网络互连。从而保证了能通过帧中继来实现高速远程互连。

帧中继交换的特点可归纳为:

(1) 帧中继交换是工作在开放系统互连基准(参考)模型的物理层和数据链路层,实现的是以“帧”为基本数据单元的交换;

(2) 帧中继交换取消了在 X.25 分组交换中所必须的复杂的差错控制,只在数据链路层进行差错检验,差错纠正则留给高层去完成;

(3) 在帧中继交换中,路由和交换是数据链路的功能,采用永久虚电路(PVC),交换的是“帧”而不是“分组”;

(4) 流量控制是通过设置帧地址域中前向显式拥塞控制标志位(FECN)和后向显式拥塞扩展标志位(BECN)来实现的;

(5) 易于其他局域网的接入。

以帧中继交换技术设备为核心组建的数据交换网称为帧中继数据网。它具有较高的吞吐量,能够提供 1.544 Mb/s(或 2 Mb/s)的传输速率,甚至进一步提供 45 Mb/s(或 34 Mb/s)的传输速率。它的延迟也很小,帧中继交换机的延迟可小于 2 ms,而 X.25 分组网的则可达 5~10 ms。

由于帧中继网的这些特性,把它和 X.25 分组网结合,作为 X.25 分组网的中继网,可以大大提高整个网络的吞吐能力,降低网络延迟。借助帧中继也能实现局域网之间高速远程互连。帧中继网已成为广域网的主干网的一种合适的选择。

参考文献

1. 杜治龙. 分组交换工程. 北京: 人民邮电出版社, 1993
2. Behrouz Forouzan et al. 数据通信与网络. 潘乞, 朱丹宇, 周正康译. 北京: 机械工业出版社, 2000
3. Sidnie Feit. Wide Area High Speed Network. Macmillan Technical Publisher, 1999 (史美林)

zhichi guocheng

支持过程 (supporting process) 有关各方按他们支持的目标负责的一系列相关活动。支持过程有助于系统或软件产品质量的提高,有助于它们的顺利运行。这类软件过程可被其他类软件过程或本类中的其他软件过程所使用。软件过程的各活动均可使用支持过程。支持过程可由使用它们的机构来实施;或作为一种服务,由一个独立的机构来实施;也可作为项目的一项规定内容,由客户来实施。一个支持过程中的活动,由实施该过程的机构负责。这类软件过程包括:文档过程、配置管理过程、质量保证过程、验证过程、确认过程、联合评审过程、审计过程、问题解决过程等。

文档过程 一个记录由某一过程或活动所产生的信息的过程。这一过程的作用是计划、设计、开发、制作、编辑、发行和维护各类文档。这些文档为系统或软件管理者、工程师以及用户等所必需。文档过程由以下一些活动构成:①过程的准备与实施,②设计与开发,③制作与发行,④维护。

配置管理过程 一个应用配置管理与技术步骤来完成下列工作的过程。这些工作包括:确定、定义一个系统中的软件配置项和基线;控制配置项的修改与交付;记录和报告配置项的完成情况和修改请求;保证配置项的完整性、相容性和正确性;以及控制配置项的存储、处理和提交。配置管理过程由以下一些活动构成:①过程的准备与实施,②配置的确定,③配置的控制,④配置情况报告,⑤配置的评价,⑥发行管理与提交。

质量保证过程 一个为使软件过程和软件产品符合所规定的需求,并按所制订的计划完成提供适当保证的过程。为了避免产生偏见,实施质量保证的人员不能是直接负责软件开发的人员,并应在组织上给予独立的权限。质量保证过程由以下一些活动构成:①过程的准备与实施,②软件产品的质量,③软件过程的质量保证。

验证过程 验证过程的目的是确定一个系统或

软件的需求是否完备和正确,以及每一阶段的软件产品是否达到了前面各阶段对它提出的要求或条件。验证可以和使用它的过程(如供应过程、开发过程、运行过程或维护过程)结合在一起实施,也可由一个独立的机构来实施。验证过程由以下一些活动构成:①过程的准备与实施,②验证。

确认过程 确认过程的目的是确定需求和最终建成的系统或软件是否满足原计划的特定应用。确认可由一个独立于供应人员、开发人员、操作人员或维护人员的机构来执行。确认过程由以下一些活动构成:①过程的准备与实施,②确认。

联合评审过程 联合评审过程的目的是评价项目的某个活动或阶段的执行情况和产品是否合适。它可以由任意二个合作伙伴所使用,由其中的一方评审另一方。联合评审过程由以下一些活动构成:①过程的准备与实施,②项目管理评审,③技术评审。

审计过程 审计过程的目的是确定遵照需求、计划和合同的程度。审计可由任何二个合作伙伴使用,由其中一方审计另一方的软件产品或活动。审计过程由以下一些活动构成:①过程的准备与实施,②审计。

问题解决过程 一个用于分析和排除在开发、运行、维护或其他过程中发现的问题或不一致(不管其性质和来源)的过程。其目的是提供一种适时的、可信赖的并编成文档的手段,以保证分析和排除所有的问题并指明各种倾向。问题解决过程由以下一些活动构成:①过程的准备与实施,②问题解决。

参考文献

1. IEEE Standard for Developing Software Life Cycle Processes — IEEE Std. 1074 ~ 1991
 2. ISO /IEC 12207:1995 Information Technology — Software — Part 1: Software Life - Cycle Process
- (黄嘉启)

zhishi biaooshi

知识表示 (knowledge representation) 描述思考内涵的抽象替代体(包括现实世界在思考过程的映象及推理的说明)。这个抽象替代体是进行有效计算与人类表示实际世界问题的中介物。知识表示是人工智能研究的基本问题。

在人工智能中,经常使用的知识表示方法列于表1。

表1 各种知识表示方法

知识表示方法	科学来源
逻辑表示	哲学家与数学家对智能行为中演绎推理作用的研究
产生式系统表示	心理学对智能行为激励—反应现象的刻画
语义网络表示	心理学对记忆的研究
框架表示	对人类感知现象的研究
过程表示	对智能系统中控制策略刻画的需要
基因表示	生物学对遗传与进化的研究
连接机制表示	神经科学研究
直接表示(拟真)	对人类表述问题的分析与观察

这些知识表示方法几乎均是来源于研究者对智能行为在微观与宏观的不同科学层次之观察与分析所抽象出的模型。对于这些知识表示方法的细节及它们的优缺点可以参见其他有关词条。

无论对人还是对机器,智能的关键性质是不确定性与灵活性。因此,如果将这些知识表示方法理解为对智能行为建模所使用的原型,则可以根据它们描述智能行为的不确定性与灵活性的程度,为这些表示方法建立一个体系树。在表1中列出的知识表示方法,根据其知识表示原理可以分成下述三类:

局部表示类 包括逻辑、产生式系统、语义网络、框架及过程表示;

分布表示类 包括连接机制表示及基因表示;

直接表示类 包括图示、图像及声音等的直接表示。

所谓局部表示就是指在系统中,这种知识表示方法所规定的任何局部(符号)具有独立的含意,它们不随系统的其他部分的改变而改变。在人工智能中,这类表示方法一般也称为物理符号机制。

采用逻辑作为刻画智能行为模型的原因在于假设推理是智能的核心,推理是区别智慧与本能的标志。在人工智能中,传统逻辑表示,一般是指一阶谓词原型,其优点是这种知识表示有坚实的数学基础,它保证了推演的保真性及完备性;另外,这种知识表示方法很容易在计算机上实现。但是,它对智能行为的描述,需要满足所建立的模型是对世界某个部分的真实描述的条件,这样才能保证基于逻辑方法所推演的结果与实际现象相一致。这意味着采用这种知识表示方法的任一系统,如果对世界的描述存

在误差,将可能导致错误甚至荒唐的结果。此外,一般来说,这种知识表示方法还将遇到计算的复杂性。

与逻辑表示方法最相近的是产生式系统表示方法(参见产生式表示),但是这种知识表示方法无论是从科学的观察上还是形式化基础上均与逻辑表示不同。心理学家发现智能行为中的激励—反应现象与产生式原理相一致。这种知识表示原型借用了逻辑蕴含的操作,但作了适合描述激励—反应现象的转义——符号替代。从形式化角度来讲,在不考虑不确定因子的情况下,这种知识表示方法是保真的,也是完备的。一般地说,在人工智能研究中,为了减少搜索空间或描述可信程度,基于产生式的系统均需要考虑不确定推理,其实质是放弃了逻辑意义下的蕴含操作,而给其以新的解释,这时它不仅是完备的而且是不保真的。

语义网络与框架两种知识表示方法产生在不同的年代。在 20 世纪 70 年代末,它们之间的类同性被说明。这两种方法均适于描述那些具有组织结构的知识,它在“一般”与“特殊”之间建立关系,从而,任何对知识特殊性的描述均继承其一般性描述的性质。继承性是这类表示方法的重要特性,并将其作为推理的主要机制。这个特性不仅对人工智能研究起着重要的作用,而且为软件工程研究开创了一种新的程序设计风格——面向对象的程序设计风格。这可以从 Bobrow 等人的 KRL 到 SMALLTALK,直到现在的 Flavor 及 C++ 的演变历史得到证明。当然,这里强调的是“风格”,而不是语言形式。框架表示方法还有其特殊的贡献:其一是“省缺”概念,它已成为常识知识表示的重要研究对象;另一个是从框架发展出的脚本表示方法,它可以描述事件及时间顺序,并成为基于示例推理(CBR)的基础之一。

过程表示方法是人类历史上使用最悠久的传统知识表示方法。自有计算机以后它成了程序设计语言中不可或缺的部分。基于描述智能行为的需要,过程表示中又加进了许多智能特性。在人工智能历史上关于知识是过程的还是陈述的,有过一段有趣的争论,其争论的结果是,在人工智能的系统中需要集成过程知识与陈述知识。

以上介绍的是在人工智能中的几种最基本的局部表示方法,它们构成了知识表示体系树中的一个分枝(见图 1)。这些知识表示方法的共同特点是要求满足“知识显现表示”的条件。换句话说,在系统中,无论是符号还是符号之间的关系均需要显现地说明,即使不确定因子也不例外。这些不确定因子

需要具有独立的含意:或是一种序或是某种统计性质,这个特性对于表示那些感知信息将产生困难。为此,在 60 年代被人工智能抛弃了的感知机思想近几年在解决了其学习算法的困难之后又重新回到人工智能的研究领域。从知识表示角度讲,由于使用了知识分布在网络之中的特性,因此相对于局部表示,人们称其为分布表示。

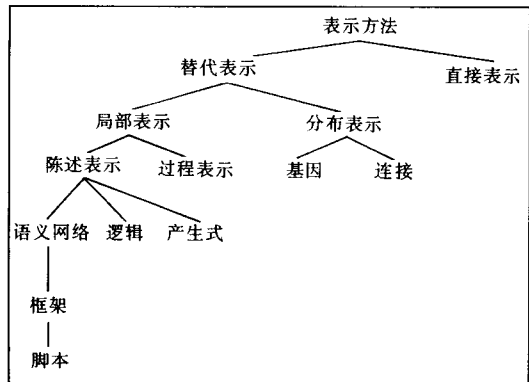


图 1 知识表示的体系树

分布表示是考虑对知识的表示取决于一组互相关联的数值,它们当中的任一个对所表示的知识均无独立的意义。具体地说,知识被分布地表示在这一组数值之中。

基因表示方法是一种近几年才被人工智能研究者认真考虑的一类介于局部与分布表示之间的知识表示方法。它的分布性表现在这种知识表示方法的基本单元染色体的任一个基因与所表示的知识没有任何直接的对应关系,只有一段基因的合理组合才具有一定的含意,因此,可以认为知识是分布地表示在染色体的某段基因之中;而局部性主要考虑到染色体可以被分成若干有实际含意的基因段。尽管从对染色体上的遗传操作来看,知识表示是分布的;但从其对后代的选择来看,知识表示又是局部的。对人工智能研究来说,基因表示适合表示那些具有整体特性的知识。另外,这种知识表示方法所基于优化的搜索算法具有大规模并行处理(MPP)的特点,因此,它对解决很多优化问题具有特殊的意义。

最典型的分布表示方法是连接机制表示,更具体地说,就是以人工神经网络作为原型的知识表示方法。在人工智能研究中,这种知识表示方法可以说历经磨难。它的出现早于许多其他知识表示方法将近二十年,但在 20 世纪 60 年代末,由于寻找其有效学习算法的失败及对其泛化能力的怀疑,导致这

种知识表示方法退出了人工智能的研究舞台,直到 80 年代初,人工智能研究者才将这类知识表示方法的原理以连接机制这个术语重新纳入人工智能研究的范围。当然使其在智能研究中占有一席之地的直接原因乃是 BP 算法的出现。尽管 BP 算法的成功是使其引起人工智能研究重新注意的原因,但这不是全部,更重要的是这种知识表示方法可以刻画那些使用局部表示难以描述的与感知类似的知识。例如,对于某些评价问题,领域的专家可以对他们熟悉的领域中的实例进行正确的评价,但是这些专家却难以解释对某个实例作出评价的理由,无论是统计意义下的理由,还是机理上的理由,而在多因素情况下甚至难以对这些因素进行排序,这就导致了使用局部表示的困难,即,大量的例外使系统难以承受,这就是知识的组合爆炸问题。从优化的角度来看,如果将实例理解为超平面上的一些点,原则上可以使用逼近理论来解决这个困难,但问题是什么样的数学基函数既可以给出在智能意义下的科学解释而又存在方便的数学处理方法呢。BP 算法的贡献就是找到了符合这两个条件的数学基函数,而这个数学基函数就是连接机制的知识表示方法。对这类知识表示方法除了上述的优点之外,并行性也是其重要的特性。但是,这种知识表示方法也同样存在显著的缺点,主要是学习收敛速度及泛化能力问题。前者无需多说,而后者则是由于这种表示所定义的数学空间与自然界中的问题空间之间存在巨大差异所造成的,而且这个差异将随问题规模的增加呈指数规律上升,其泛化能力则呈指数规律下降。目前,从理论上还未找到有效的解决方法,而在技术上则采用与其他局部表示集成的方法来克服这个困难。基因表示与连接机制表示构成了知识表示体系树中的另一分支。

采用与自然界一致的知识表示方法早在 60 年代初就已被提出,现在,这类表示方法被人工智能研究者称为直接表示或拟真表示,如地图、图形甚至音乐等,它们以自然的方式表示了关于世界上知识的某些方面。基于这类知识表示的系统是以对实体的拟真描述直接或间接参与推理为特点。

如果以计算机对知识的编码作为讨论的条件,则直接表示类不是一个可以完全独立于局部与分布表示类的方法。主要原因是考虑到任何知识表示方法必须可被计算机所接受这个先决条件。因此直接表示类所采用的方法需要变换为局部或分布表示的形式。对计算机而言,相对于局部与分布表示,直接

表示可以称为外部表示。直接表示与其他内部表示相比较,它们不在一个层次上,它强调知识表示与表示实体间具有结构相似性。因此,在人工智能中,对这类知识表示方法的研究具有特殊的意义。

直接表示方法在较长的时间内没有得到长足发展,其主要原因是:①计算机对直接表示的信息难以处理,表现在直接表示的信息(例如,图形)具有很强的领域相关性,暗示这种知识表示方法包含了太多冗余信息,因此聚焦则成为必须考虑的问题。另外,大多数直接知识表示的信息语义取决于其使用的背景而不独立,这样,直接表示难以成为一种一般性的描述语言;②直接表示难以表示定量信息,换句话说,直接表示不能完全描述自然世界的全部信息,这使很多研究者试图设计使用直接表示的语言均以失败而告终。

近几年,直接表示又得到人工智能研究者的重视,但其研究课题已大大被扩展:①研究者已认识到在纯粹的直接表示中所存在的缺点不可能通过其自身的改善而得到克服,它需要利用其他媒体表示的方法去补充直接表示的不足,这些研究将被发展成为多种媒体表示的人工智能;②直接表示的另一个引申是研究所谓现场智能与虚拟现实。这两种方法均是以世界上实际对象作为表示原型,特别是现场智能,它甚至不考虑外部世界到内部知识表示转换过程的细节。一般地说,直接表示描述问题的缺陷是其不具有表示知识的一般性。因此,不采用多种知识表示方法构成互补是难以解决这个难题的。

上面讨论的几类知识表示方法,可以总结为图 1。

在这个知识表示的体系树中,局部表示是人工智能研究最充分也是正统人工智能最经常使用的知识表示方法。分布表示方法应该说是局部表示方法在智能行为描述上不够充分的补充。而直接表示,尽管其产生的年代几乎与局部表示的各种方法同时,但其发展却十分缓慢。目前,直接表示引起了越来越多的人工智能研究者的注意。主要原因除了直接表示所特有的优点之外,它还具有:①出于对界面的考虑,人们越来越不能容忍计算机只能接受计算机学者所规定的知识表示方法及只有人工智能研究者才能解释的机器智能行为的事实,而需要以更接近人所习惯的通信方式与计算机交互,并能够以非人工智能研究者就可以了解的方式解释其行为;②人工智能已发展形成的一些理论与方法可以很好地支持直接表示方法,例如,基于示例推理的技

术就可以作为直接表示方法的研究工具;③媒体技术与虚拟现实技术的发展,为直接表示的研究开辟了模式识别、图形学及人工智能的集成之路。

总之,尽管人工智能发展的历史已经提出了大量得到实践考验的有益的知识表示方法,但是,知识表示问题并未真正解决。近几年,研究者正在对表示问题的本质进行更深入的讨论,由此形成了各种阐述研究者对智能行为研究的方法论及对其本质认识的表示观。

根据人工智能研究,特别是在专家系统研究中所存在的系统脆弱性等问题,人们认识到其根本原因是常识知识问题未能得到解决,当研究者试图解决这个问题时,对人工智能研究,特别是对知识表示的研究产生了深刻的变化。“常识知识存在简洁的表示模型吗?”“人工智能需要这类表示常识知识的简洁模型吗?”“如果这类简洁的表示模型存在,它可计算吗?”对这些问题的回答将要涉及“什么是知识表示”这一基本问题。根据对这个基本问题的不同理解及所采用的不同方法论,人工智能学界形成了不同的学派,对表示而言,可以称为“表示观”。

认识论表示观假设:知识表示是对自然世界的描述,其惟一的作用就是携带知识内容,这意味着知识表示可以独立于推理与搜索来研究,知识表示自身不显示任何智能行为。表示研究与启发式研究无关。

本体论表示观假设:知识表示是对自然世界的一种近似,它规定了看待自然世界的方式,即一个约定的集合。这意味着,对自然世界而言,基于本体论表示观,表示只是描述了在这个世界中,观察者当前所关心的那部分,其他部分则被忽略。

知识工程表示观假设:知识表示是计算机对自然世界描述的模型,它应该满足计算机这一实体的具体限制,因此,知识表示可以理解为一类数据结构及在其上的一组操作。

不同的表示观将决定智能模拟研究所采用的策略及机器智能行为的评价标准。例如,知识工程表示观强调自然世界在计算机内部某类数据结构映射形式及对存储于其中的内容所采用的处理方法,因此,研究知识的存储结构及对它的有效使用(推理)成为这种表示观研究的主要任务;认识论表示观认为,知识表示是一种携带知识的理论,至于它是否可计算甚至存在着一个算法均不是重要的,特别是问题求解的有效性完全不在其考虑之列;本体论的知识表示观则认为任何知识表示均是不完全的知

识理论,而对其使用的有效性(计算困难程度)则是先决条件,因此,本体论的表示观强调一种聚焦的功能,即认为观察者在任一瞬间所关心的事物仅仅是自然世界的一部分,而世界的其他部分则被忽略。这个知识表示观建议将知识表示与计算困难程度联系起来考虑,因此,推理与搜索成为知识表示问题研究的一部分。一般地说,认识论表示观强调某种存在性的研究,本体论表示观则更多考虑构造性的研究,而知识工程表示观则以可实现性作为重点。显然,对任一门学科,存在性、构造性及可实现性均是重要的,简单地否定某种表示观是不合适的,甚至是错误的。

参考文献

1. Barr A, Feigenbaum E A. The Handbook of Artificial Intelligence, Vol. 1. Pitman, 1981
2. Ginsberg M L. Reading in Nonmonotonic Reasoning. Morgan Kaufmann Publishers, Inc., 1987
3. Davis R, Shrobe H and Szolovits P. What is a Knowledge Representation? AI Magazine, Spring 1993

(王珏 石统一)

zhishi gongcheng

知识工程 (knowledge engineering) 研究知识的处理,并提供研制基于知识的智能系统的一般方法和基本工具的学科。知识工程是人工智能、数据库、数理逻辑、认知科学和心理学等学科交叉发展的结果。知识工程的研究使人工智能学科发生了重大改变,从探索广泛普遍的思维规律转向利用知识解决特定问题的研究。

知识工程主要研究知识获取、知识表示、推理策略,以及开发方法和环境。为了使计算机能运用专家的知识处理问题,首先要获取知识,包括经验知识和书本知识,来建立知识库,进而采用一定的形式表示知识。目前常用的知识表示方式有产生式规则、语义网络、框架、状态空间、逻辑模式、脚本、过程、面向对象等。然后,通过推理来求解问题。推理方法通常有演绎推理、不确定推理、非单调推理等。

知识工程的概念是美国斯坦福大学的 Feigenbaum 于 1977 年首先提出的。知识工程的研究促进了知识系统的发展。知识系统包括专家系统、知识库系统、智能决策系统等。知识工程中所研究的最典型的智能系统是专家系统。它实现了人工智能从理论研究走向实际应用、从一般推理策略探讨转向运用专门知识的重大突破。20 世纪 60 年代初,出

现了运用逻辑学和模拟心理活动的一些通用问题求解程序,它们可以证明定理和进行逻辑推理。但是这些通用方法无法解决大的实际问题,很难把实际问题改造成适合于计算机解决的形式,并且对于解题所需的巨大的搜索空间也难于处理。人们解决实际问题并不全靠推理,而常用“半逻辑”方法,即一些不精确的和不确定经验规则,专家正是大量运用这些知识作出有用的结论。1965年,费根鲍姆等人在总结通用问题求解系统的成功与失败经验的基础上,结合化学领域的专门知识,研制了世界上第一个专家系统 DENDRAL,可以推断化学分子结构。30多年来,专家系统的技术不断发展,应用渗透到众多领域,其中不少在实际应用中产生了经济效益。

知识库系统(KBS)是把知识以一定的结构存入计算机,进行知识的管理和问题求解,实现知识的共享。知识库系统的明显特色是将推理和查询结合起来,改善知识库的维护功能,为开发具体领域的知识系统提供有用的环境。今后知识库系统研究的重点是知识模型、知识库模式、知识操纵语言、推理机与数据库耦合等问题。

知识工程逐步应用于**决策支持系统(DSS)**,产生了智能化的DSS。由数据库、模型库、知识库、图形库和文本库组成的智能决策系统,将为解决半结构化、非结构化的决策问题提供有效的手段,提高科学管理的水平。

为了提高知识系统开发的效率和质量,知识工程特别重视知识工程环境和工具的研制。典型的系统有美国的EMYCIN、知识工程环境系统KEE、CYC,中国的面向对象知识处理系统OKPS、天马系统等。开发工具有通用的推理机、**知识获取工具**、解释子系统、用户界面等。在知识工程环境或工具支持下,开发特定应用领域的知识系统的主要工作是建造知识库,促进知识工程应用推广。(史忠植)

zhishi huoqu

知识获取(knowledge acquisition) 将用于问题求解的专门知识从某种知识源(如人类专家、文本、数据)中总结和抽取出来,转换为一种形式化的知识的过程。

知识获取是建立知识系统的关键,同时也是一个耗时、低效的过程。目前尚缺乏一种统一有效的知识获取方法。因而,知识获取被认为是知识处理中的一个“瓶颈”。事实上,有许多专家知识是无意识的,是难以直接用语言表达的。因此,知识获取并

不是简单地将知识从一种表示形式转换为另一种表示形式的过程,而是一个为专家知识建立相应问题求解模型的过程。这种过程并不能一次完成,而是贯穿于知识系统开发和维护的整个过程。由于知识本身的多样性,目前尚无一种统一的知识获取方法。知识获取所要研究的问题涉及以下两个方面:

(1) 对知识源进行泛化、选择、分类和组织;

(2) 对知识的求精、检查,保持知识的一致性、完全性和无冗余性。

知识获取的三种方式 知识获取可分为人工知识获取,半自动知识获取和自动知识获取三种方式。

人工知识获取 知识工程师作为中介人,沟通知识源(主要指领域专家)和计算机系统,由知识工程师抽取、组织知识和实现编码。采用的主要技术是与专家的会谈技术。由于知识工程师对领域缺乏了解,使专家资源缺乏,加以交流过程产生误差造成了人工知识获取是一个费时、低效且不精确的过程。

半自动知识获取 借助于**知识获取工具**的帮助完成知识获取过程,但知识工程师的干预仍是不可缺少的。它又称为交互式知识获取。半自动知识获取利用知识获取工具使知识工程师从人工知识获取的工作中部分地解放出来。这种获取方式的主要问题之一是构造工具的代价和使用工具的得益之间的平衡问题。

自动知识获取 获取过程完全由知识处理系统自动完成。目前主要指传统的机器学习过程。

人工知识获取尽管耗时和低效,但仍是目前建造**专家系统**中最常用的知识获取方法。随着知识获取技术的发展,已有不少实用的知识获取工具问世,但由于知识类型的多样性和复杂性,这些工具的应用受到了很大限制。自动知识获取工具目前还处于研制阶段。关于半自动知识获取和自动知识获取的更详尽的内容请参见**知识获取工具**。

知识库系统中的知识获取涉及知识库系统的整个生命周期,所涉及的阶段可分为:

(1) 问题识别阶段 确定参与人员、资源、系统设计目标和问题主要特征等;

(2) 概念化阶段 确定领域基本概念及其关系,对系统目标进一步分解等;

(3) 形式化阶段 选择合适的知识表示方式,将基本概念、关系、子任务等表示出来,确定相应**数据结构**和问题求解模式;

(4) 实现阶段 将形式化阶段得出的知识映射成可执行的程序,实现一个知识基系统原型。知识

工程师的任务是通过原型系统的运行,评价所选取的**知识表示**的合适性;

(5) **测试阶段** 选用更多、更全面的例子从整体(表示、推理、结构)上全面评价原型系统。对不合适部分进行修改。

以上五个阶段是一个不断反复的过程。

人工知识获取的主要技术 知识获取中采用的许多主要技术并不是专家系统研究者们所独创的,而是已在其他学科领域(如心理学、软件工程、系统分析等)中被采用过的。

会谈技术 知识获取中最原始也是最基本的技术。通过会谈,知识工程师可以快速掌握重要的领域概念和词汇。这项技术主要应用于知识获取的早期阶段,一般用于获取说明性知识。

会谈分为非结构性会谈和结构性会谈两种形式。

非结构性会谈的内容比较随意,没有一定指向性和系统性。它一般适用于准备拟定议题的初始阶段。

结构性会谈有一定目的和议题,有严格的时间安排和问题清单。是一种有计划、有组织、系统性的会谈。它一般适合于知识工程师探讨问题的细节,并产生具体、有用的知识。

会谈还可作为知识工程师和领域专家之间的知识渗透、情感交流、相互理解、进而加速知识获取进程的一种手段。

会谈中需注意以下问题:

(1) **会谈的记录** 每次会谈的详细记录都要保留下来,包括磁带录音和由记录产生的文本。

(2) **有效地利用问题** 会谈的基本形式是提问和解答。会谈的效率取决于问题的适当性和解答的清晰性。对知识工程师来说要把握住提问问题的形式、层次和次序。

(3) **会谈的总结** 知识工程师需要将会谈记录整理成文本形式,与领域问题相关的术语、词汇要在文本中定义。同时,将整理过的文本让专家过目,以确定无误。

决策分析 跟踪专家具体的决策过程,了解专家在决策时考虑了哪些因素及如何运用这些因素和有关知识。对专家来说,仅通过会谈难以直截了当地说出决策因素和过程。若以实际案例为线索,评理解处理知识和手段,更易使专家将注意力集中于一系列案例而不转移主题,从而使评解容易连贯和形成一个结构。

草案分析 一种决策分析技术,让专家将决策过程说出来(有声思维),知识工程师做记录,形成专家口头报告的记录文本(草案)。进而分析草案、识别决策因素、决策顺序。

这种决策分析技术比较适合于获取过程性知识或专家无意识中的启发性知识。

对范例的选择有下述几种方法:

(1) **约束条件** 简化问题的条件,以便揭示主要决策过程。

(2) **约束解** 为节省时间和考虑主要决策因素,在求解问题时,只产生能反映获取目标的局部解,而不做深入追究。

(3) **场景模拟** 对于无法在实际现场进行追踪的问题,进行场景模拟。

(4) **情节类比** 从一个已知的案例中,类推出另一个案例,进行类比分析。

(5) **分析难题** 寻找一些有一定难度的案例进行分析,有助于了解真正的专家知识。

概念组织 分析、整理和组织基本概念,并发现基本概念间的相互关系。获取概念的基本方式是会谈,进一步的概念组织涉及下述各项技术:

(1) **概念字典** 将基本概念按其相关性进行分组,并对各组给出名词标识(相当于字典索引)。一般以图示方式表示。

(2) **分类卡** 将各概念或对象写在卡片上,帮助组织知识。

(3) **概念分类** 根据概念间的相关性将概念组织成一个树形层次结构。

(4) **心理量化** 以量化形式表示各概念,分析概念间的关系。分为多维度量化,层次结构聚类分析和网络量化三种。

储存格技术 一个抽取和分析领域专家模型的技术。起源于心理学学者 George Kelly 的个人构造理论 PCT。储存格技术采用储存格表示人类专家对事物(概念)的认知模型。储存格是一个由元素(对应于例子)和构造(对应于属性)组成的二维矩阵。矩阵中的每项值是专家对相应元素所具有的相应构造量的评价。储存格技术包括两个过程:①初始储存格的产生涉及知识工程师与专家交谈,不断选定用于评价元素的构造和相应评价。②分析储存格主要通过储存格中的量化评价,分析元素间的相似性和构造间的相似性,同时进行排序。根据分析结果,专家可以修改初始的储存格。

上述两个过程不断重复进行,直到专家对储存

格所反映的模型感到满意为止。

储存格技术比较适合于抽取凭直觉感知的概念。

多专家知识获取 对于一些复杂的,涉及不同学科知识的问题往往需要从多个专家那儿获取知识。各专家对同一问题的看法有时不尽一致,这就涉及到多专家合作的问题。下面是几种常见的多专家合作技术。

为了获取不同专家对领域问题的共同看法和分歧点,可采用以下几种多专家会谈技术:①头脑风暴——自由发言,快速产生大量思想而不做评价;②一致决策——多专家对可能的见解和方案共同进行评价和权衡,以达成一致;③虚组技术——允许各专家单独或匿名地发表自己的见解,不受他人干扰。

另外,还可以利用心理量化、归纳学习或储存格技术,分别从各专家处获取相应知识的组织和分类方式,然后比较不同专家对领域理解的共同点和不同点。

对于一些反映专家间不同看法的知识(或规则),可在知识系统中引入无规则、不确定推理等技术加以综合处理。

发展趋势 知识获取作为知识工程的关键分支之一,经过近二十年的发展,取得了一定进展。今后的发展趋势有下述几个方面。

(1) 知识获取方法论和基础 现已有众多的开发方法和原则。如何在统一的理论和一致的框架下,讨论集成各系统、实例和研究方法,是一个迫切需要解决的问题;如何在这一致的框架下,提出系统化的评价方法与原则,从而达到鉴别不同研究方法和技术的优劣。

(2) 知识获取系统的体系结构 探讨基于模型制导的知识获取的体系结构,是今后一大趋势,并且连同使用元级体系结构和自省型体系结构来组织知识获取的任务,以达到高效率的知识处理的目的。协商式知识获取的结构,随着分布处理与网络技术的发展,越加显示出重要性。

(3) 知识获取的表达 采用更加智能化的图形表达机制与适当的中间表达机制,是一种重要的研究方向;考虑在不同级别上的知识表达的集成,有利于知识获取在不同级别上的知识保真;领域模型的更加精确的规格描述;本体论、策略知识等不同知识类型的有效描述的表达,亦将是今后的一大趋势。

(4) 知识获取的自动化 为了提高知识获取的

效率和准确性,降低人工代价,有必要让知识获取自动化。机器学习是知识获取自动化的关键技术。机器学习技术不仅使得从专家提供的例子和数据中归纳有用知识成为可能,而且使得通过学习不断完善知识成为可能。有人断言,机器学习可能是解决专家系统中知识获取这一瓶颈问题的关键。

参考文献

1. McGraw K. Knowledge Acquisition: Principles & Guidelines. Prentice Hall, 1989
2. Kidd A L. Knowledge Acquisition for Expert System: A Practical Handbook. Plenum Press, 1987
3. Boose J H and Gaines B R. The Foundation of Knowledge Acquisition. Academic Press, 1990

(王申康)

zhishi huoqu gongju

知识获取工具 (knowledge acquisition tool)

帮助领域专家或知识工程师实现知识抽取和转换的一种软件系统。其目的是为了取代或部分取代知识工程师在知识获取过程中的作用。知识获取工具的帮助既可以涉及知识获取的全过程(如与领域专家交互、领域概念化、建立模型、知识维护等),也可以只涉及知识获取的某一过程。由于领域问题的广泛性和复杂性,目前尚无一个通用的涉及知识获取全过程的知识获取工具。

知识获取工具的应用可以给知识获取过程带来以下几点好处:

- (1) 可以降低知识系统开发过程中人力资源的代价;
- (2) 由于工具中特定知识获取技术的引入,可提高知识获取过程的标准化程度;
- (3) 减少由于过多人员的干预而产生的误差,提高知识库的质量;
- (4) 专供领域专家使用的工具,可提高领域专家在时间上的自主性和提供知识的兴致。

1968年,J. McCarthy的“Advice taker”可算是最早的一个体现知识获取工具思想的系统。在这以后,人们相继开发了一些智能编辑系统,基于心理学会谈方法的系统和利用自然语言理解技术与领域专家直接对话的知识获取工具。到了20世纪80年代,由于机器学习技术的成熟,相应的学习技术也逐步应用到知识获取中来。近年来,人们相继开发了一些知识获取工作平台,以辅助领域专家实现知识概念化、知识表示、知识建模和知识维护等。知识工

程师则作为工作平台和领域专家之间的协调者。

就目前研究状况来说,知识获取过程还没有形成一种公认的方法。研究者们在不同领域里分别开发出了一些知识获取工具,要对这些工具进行准确分类和比较是困难的。一般来说,可按以下几种分类标准对知识获取工具进行分类:

(1) 按工具的应用领域特性进行分类 如,工具是否独立于应用领域(即工具的通用性),相应领域的任务类型,相应的问题求解方法(参见启发式搜索)类型。W. Clancey 等人提出了将领域任务类

型分为:分析类型、综合类型及它们的组合三种。这些类型还可继续分为若干子类:①分析类型可分为识别、诊断、纠错、解释等类型;②综合类型可分为组装、评价、设计、规划、调度等类型;③分析综合类型可分为控制、指导、监控、预测、修补等类型。不同的任务类型还与问题求解方法有联系。这里指的问题求解方法分为两大类:启发式分类方法和启发式构造方法。

图1表示了不同知识获取工具与所适用的任务类型和问题求解方法的关系。

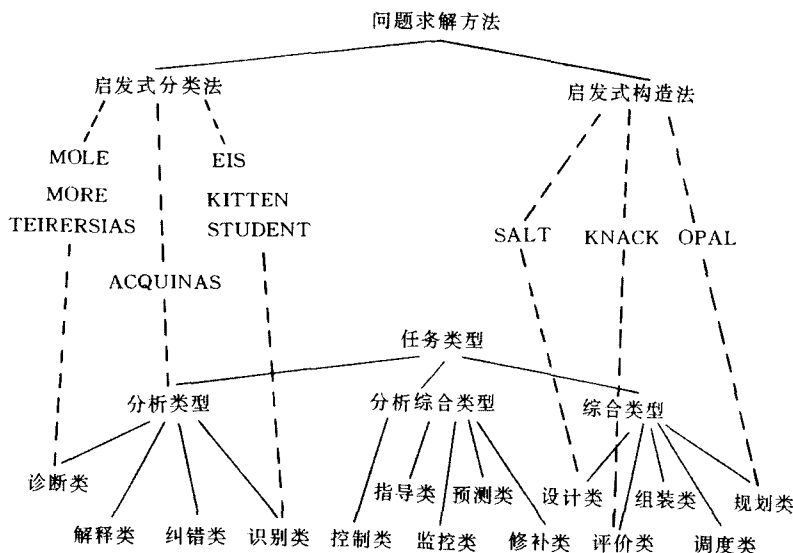


图1 工具、任务类型和问题求解方法关系图

(2) 按知识表达方式分类 专家知识在计算机内可有不同的表达方式(如规则、关系、表、层次结构、储存格),同时,要获取的知识类型也可能不同(如概念、目标、范例、过程性知识,策略性知识等)。

(3) 按工具性能分类 ①系统的自动化程度,一般分为半自动化(交互式)和自动化(采用机器学习技术)两类;②对知识获取过程的支持程度,是全过程支持还是支持知识获取的某一或某几个阶段。

(4) 按工具采用的技术分类 这是最重要的一种分类方式。

这些分类方式并不是互斥的,即某一具体工具可能同时采用几种技术。

目前知识获取工具采用的主要技术有下述几种。

(1) 基于心理学和会谈方法 ①采用会谈技术,通过专家与工具交互获取知识,如 AQUINAS,

ASK, MOLE, MORE, SALT 等工具;②采用草案分析技术(参见学习计算理论),记录和分析专家对领域任务的评解文本,以产生一个专家知识的结构化模型,如 CODE 等工具;③采用储存格技术,利用心理学中的个人构造理论(参见学习计算理论)抽取和分析知识,如 AQUINAS, ETS, KITTEN 等工具。

(2) 基于领域模型方法 从某种角度讲,知识获取是在计算机中为专家知识建模的过程。基于领域模型方法根据领域特性和知识系统结构使知识获取过程知道抽取什么知识和怎么抽取。具体有以下几种技术:①利用领域问题求解方法模型指导知识获取,如 AQUINAS, MOLE, MORE, SALT 等工具;②利用领域结构指导知识获取,如领域因果模型(MOLE 等),概念模型(APPENTICE, CODE 等),决策模型(AXOTL 等);③利用知识建模语言定义和描述领域问题和问题求解方法。典型代表是

知识辅助文档系统 KADS。KADS 是在知识层上提出的基于知识建模的知识获取方法论。它的代表人物是 B. Wielinga 等人。KADS 的方法论包括建模原则,基于问题求解方法的建模框架和由领域描述语言 DDL,问题求解描述机制 PSM 组成的建模语言三个层次。

(3) 知识结构浏览 提供一种浏览和编辑知识结构的功能,如 APPENTICE, CYC 等工具。

(4) 多专家知识获取 从多个知识源中抽取和分析知识,并进行综合,如 AQUINAS, ETS, KITTEN 等工具。

(5) 机器学习技术 以上几种技术主要用于交互式(半自动化)的知识获取工具,而机器学习使知识获取系统能从专家的指导和示例中归纳有用的知识。目前应用较多的有基于类比学习、基于解释学习、实例学习以及数据库中知识发现等机器学习技术(参见机器学习)。

近年来,随着知识获取研究的深入发展,各式各样的知识获取工具被不断推出,在国际上有影响的知识获取工具就达一百多种。下面简要介绍两种典型的知识获取工具:基于个人构造理论的 ETS 和基于领域模型的 MOLE。

ETS 是一个应用储存格技术的知识获取工具。它通过与专家交互对话,以储存格方式收集有关信息,进而分析储存格,将有关信息转换为产生式规则形式。ETS 还提供了知识库分析、发现矛盾知识和测试知识库性能等功能。但 ETS 比较适合于分析类型的问题,且很难获取深层的领域因果关系和一些过程性知识。

MOLE 是一个基于领域模型的知识获取工具。其获取过程的推理基础是一些假设的、与领域无关的关于知识处理过程和领域世界模型(诊断类)的启发性知识。MOLE 的知识获取过程主要是决定哪些假说可能与要解释的问题有关,哪些信息可用来区分这些假说,并由此不断询问,直到这些假说可完全区分为止。

参考文献

1. Boose J H. A Survey of Knowledge Acquisition Techniques and Tools. Knowledge Acquisition, 1989, 1: 3~37
2. Boose J H and Gaines B R. The Foundation of Knowledge Acquisition. Academic Press, 1990
3. Boose J H and Gaines B R. Knowledge Acquisition Tools for Expert System. London: Academic

Press, 1988

(王中康 何钦铭)

zhishi jinghua

知识精化 (knowledge refinement) 在知识获取完成后,对已获取的知识进行提纯、验证,转变成一个高效率、高质量的知识库的过程。它是人工智能中知识获取的一个重要组成部分。

知识精化的目的在于消除知识获取过程中因问题本身或获取手段等原因引起的提取知识的冗余和冲突,提高知识库的性能和质量,特别是满足一些有高可靠要求的应用需求,如核电站反应堆、卫星的发射与控制等。在知识工程师所提供的原型知识库的基础上,进行知识精化处理,避免了机器学习要求过多的背景知识和对问题域的严格限制,实现大量知识的获取任务。

知识精化是一个较年轻的领域。自从知识获取成为知识工程的瓶颈问题以来,知识精化得到迅速发展。前期(20 世纪 70 年代中期)的知识精化基本上是作为专家系统(知识系统)中知识库的修改器。自 80 年代初,通用知识精化系统 SEEK 和 SEEK2 出现以后,知识精化才具有较完整的模型和方法。80 年代后期,人们感兴趣的问题则是使知识精化的技术和方法更加完善和成熟,并将此与其他的知识证实技术、评价技术结合起来研究。

由于在知识获取过程中所带入的各种类型错误的性质和特征不同,各系统在精化阶段所采用的精化模型和方法都有一些各自不同的特点和策略:采用阶段性调试,对系统开发的每个阶段的结果都加以论证;或采用结果校正,对在系统投入使用阶段的实际数据加以校正;或采用模拟验证方法,将新开发的系统放置于一个模拟环境中运行,以便能发现错误。

知识精化过程常常是通过适当地、准确地设计并选取典型实例,来发现系统中的错误并做修正以提高其性能。这样的系统包括:

(1) 测试实例选择 选择测试实例的简单方法是选取实例库中所有实例,或随机或按序取一实例;或根据当前精化目标,选择若干相关的实例;或依据启发式知识或元知识参与之下,选择有效的实例。

(2) 测试和分析 测试是将实例数据输给系统,让其在这些数据上运行,然后观察其运行行为,记录必要的中间结果和最终结果。为了确定系统是否正确反映了领域专家原来的目标,是否达到预期性能指标,其结论在多大程度上、是否接近专家的结

论,我们需要根据所发现的问题,分析出产生这些问题的原因,进而提出修改计划草案。对测试数据的分析,是提高知识系统的知识质量和性能的关键。

(3) 精化试验 对所发现的每一个不合要求的性能表现,能找到多个可能的原因,提出多个修改计划草案。在一定的精化原则的指导下,筛选出某一修改计划来修改知识库的内容。

(4) 修改知识库 依据(3)所提议的修改计划,修改知识库内容。然后,重新执行(1),直到知识工程师对知识库满意为止。

知识精化有如下几种模式:

(1) 手工精化模式 由人工完成测试实例的选择,结果数据的获得和分析,产生修改草案等精化任务。不依赖于或仅依赖于简单的辅助工具。例如,Prospector 方法就是采用手工精化方法。

(2) 半自动精化模式 知识精化的大部分任务可以借助于计算机完成,而一些关键抉择仍由人参与作出。如由计算机辅助设计者选择实例,计算机辅助完成其他一些较深的分析推理。人的作用是在精化的关键地方决定下一步的精化方向。此类精化模式的典型系统是 SEEK。

(3) 自动精化模式 在摆脱人的干预情况下的一种自动精化过程。精化知识库中存放的知识之一是启发式知识,这些启发式知识可能涉及有关如何选择试验草案的知识。SEEK2 是此类精化模式的典型例子。与 SEEK 相比,SEEK2 可以精化更一般的知识库,具有“自动导航能力”,它通常不需要与人交互就能完成所有基本知识的精化任务。

从技术手段来看,知识精化的方法分为三大类:①结构精化;②功能精化;③理论重检。所谓结构精化,是指利用知识库中的知识内部结构而无需或很少借助额外知识来精化专家的知识。功能精化则是利用知识库之外的额外知识,如用户的意图等,对知识库中的知识进行精化。理论重检则是利用归纳推理和基于解释机制,借助一组训练实例来提高知识库的质量,有扎实的形式化理论基础。

从实现方式来看,知识精化的方法还可以分为:①静态精化;②动态精化。前者只是依据知识库中知识的语义相容性来精化知识库中的知识;后者则是用例子进行测试,从而了解知识库中的知识性能强弱,以达到知识精化的目的。

随着被精化的知识系统的日益复杂,知识精化的技术与方法以及相应的计算机辅助精化工具越来越显示出其重要性。精化技术和方法正朝着更自

动、更有效、更通用的方向发展;各种辅助的精化工具,也朝着更实用、更高效率的方向发展。知识精化技术,一方面,作为知识获取的一个分支和有效技术,更加紧密地结合于知识获取之中;另一方面,它的一些关键技术,如知识证实技术、知识检测技术和知识测试技术,随着知识系统应用的拓广,将日益扩大和成熟,成为知识处理技术中富有生命力的技术和方法之一。

参考文献

1. Politakis P, Weiss S M. Using Empirical Analysis to Refine Expert System Knowledge Bases. AI, 1984, 22: 23 ~ 48
2. Marc Ayel, Jean-Pierre L. Validation, Verification and Test of Knowledge-based System. Wiley, 1991

(王申康 吴朝晖)

zhishi ku

知识库 (knowledge base) 针对某一(或某些)领域问题求解的需要,采用某种(或若干)知识表示方式在计算机中存储、组织、管理和使用的互相联系的知识片集合。这些知识片包括与领域相关的理论知识、事实数据,由专家经验得到的启发式知识,如某领域内有关的定义、定理和运算法则以及常识性知识等。

知识是人类智慧的结晶。知识库使基于知识的系统(或**专家系统**)具有智能性。20世纪60年代中期,E. A. Feigenbaum 等人在建造第一个专家系统的同时,也建造了第一个知识库。并不是所有具有智能的程序都拥有知识库,只是基于知识的系统才拥有知识库。现在许多应用程序都利用知识,其中有的还达到了很高的水平,但是,这些应用程序可能并不是基于知识的系统,它们也不拥有知识库。一般的应用程序与基于知识的系统之间的区别在于:一般的应用程序是把问题求解的知识隐含地编码在程序中,而基于知识的系统则将应用领域的问题求解知识显式地表达,并单独地组成一个相对独立的程序实体(见图1)。

知识库的特点如下:

(1) 知识库中的知识根据它们的应用领域特征、背景特征(获取时的背景信息)、使用特征、属性特征等而被构成便于利用的、有结构的组织形式。知识片一般是模块化的。

(2) 知识库中的知识是有层次的。最低层是“事实知识”,中间层是用来控制“事实”的知识(通

常用规则,过程等表示方法表示);最高层次是“策略”,它以中间层知识为控制对象。策略也常常被认为是规则的规则。因此知识库的基本结构是层次结构,是由其知识本身的特性所确定的。在知识库

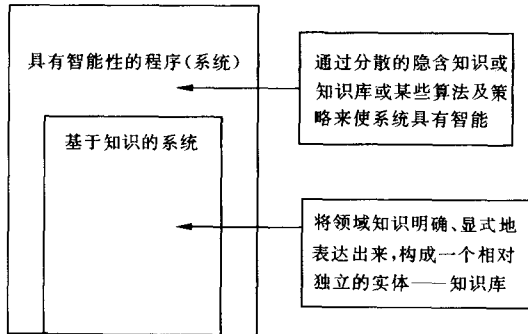


图1 知识库在基于知识的系统中的作用

中,知识片间通常都存在相互依赖关系。规则是最典型、最常用的一种知识片。下面以规则为例,说明在规则库中大量存在的这种依赖关系。对规则库首先应进行分组。知识库中的所有规则都是围绕一组领域问题来组织的,与每个领域问题相关的规则构成一个连通的规则集结构图。领域问题规则集结构图间可能存在多种关联,且关联的密切程度也不一样。

(3) 知识库中可有一种不只属于某一层(或者说在任一层次都存在)的特殊形式的知识——可信度(或称信任度,置信测度等)。对某一问题,有关事实、规则和策略都可标以可信度。这样,就形成了如图2所示的增广知识库。在数据库中不存在这种不确定性度量。因为在数据库的处理中一切都属于“确定型”的。

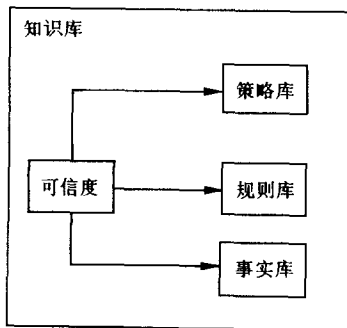


图2 增广知识库

(4) 知识库中还可存在一个通常被称作典型方法库的特殊部分。如果对于某些问题的解决途径是

肯定和必然的,就可以把其作为一部分相当肯定的问题解决途径直接存储在典型方法库中。这种宏观的存储将构成知识库的另一部分。在使用这部分时,机器推理将只限于选用典型方法库中的某一具体部分。如果加上“典型方法库”部分,则知识库将如图3所示。其中,事实库、规则库指向典型方法库的箭头表示当典型方法库中所存储的某问题的解决途径已被选中时,有关事实及规则将被调出连同该途径共同生成实际问题的解。

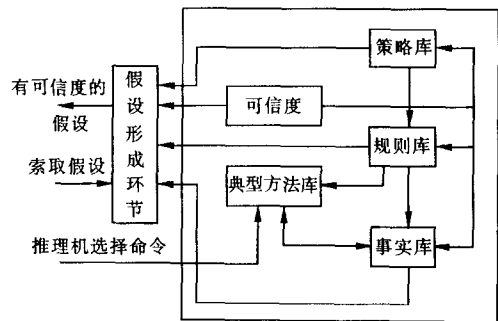


图3 一个包含典型方法库的知识库

另外,知识库也可以在分布式网络上实现。这样,就需要建造分布式知识库。建造分布式知识库的优越性有三:

- (1) 可在较低价格下构造较大的知识库;
- (2) 不同层次或不同领域的知识库对应的问题求解任务相对来说比较单纯,因而可以构成较高效的系统;
- (3) 可适于地域辽阔的地理分布。

知识库的构造必须使得其中的知识在被使用的过程中能够有效地存取和搜索;库中的知识能方便地修改和编辑;同时,对库中知识的一致性和完备性能进行检验。

参考文献

1. Frost Richard A. Introduction to knowledge base system. London: William Collins & Co. Ltd, 1986
2. 管纪文,刘大有等. 知识工程原理. 长春: 吉林大学出版社, 1988 (刘大有 郑方青)

zhijie biao shi

直接表示 (direct representation) 一种知识表示(或称拟真表示),其中表示与被表示实体间具有结构相似性。它的突出特点是实体的拟真表示直接或间接参与推理。

采用与实际世界一致的自然表示方法早在

60年代初就已被提出,这类表示方法直接地在系统中使用,像地图、模型、图形甚至音乐等拟真的自然方式用于表示关于世界的某些方面。尽管与其他表示方法存在着差别,但是直接表示的模式仍然可作为一种语言。在人工智能历史上使用直接表示的系统,均将实体的拟真表示直接或间接参与推理,例如 Gelernter 的基于传统的欧氏几何证明方法的几何定理证明器,它的输入是对前提和目标的陈述以及用一系列坐标来表示的图示,在证明过程中证明器把图示作为启发式信息,排除在图示中不正确的子目标,从而大大地减少了搜索。

在设计实际系统时,尽管直接表示可以使用类似谓词表示的**数据结构**,但对它们的处理是不同的,即数据结构的性质将以不同方式应用于推理中。例如,如果一个查地图的程序仅从内部数据结构中搜索出距离,而不是直接读图,就不能说这个地图对于距离来说是直接的。直接表示对某些信息是直接的,对另一些信息则不是。

直接表示的优点是:①它可以使一些重要信息通过“观察”而获得,从而免去了用谓词表示所需的复杂的推理;②与逻辑表示相比,直接表示适合于较少的模型,从而使问题求解的效率更高;③直接表示可以减少搜索。但是,直接表示又有以下缺点:①在某些情况下,概括是必须的,而直接表示概括能力差;②当直接表示的某些特性与实际情况不符时,不易发现这些特性;③直接表示不能描述某些定量信息。因此,对直接表示的应用必须考虑所需要解决的问题是否与这种表示方法的优点相一致。

参考文献

Barr A, Cohen P R and Feigenbaum E A. The Handbook of Artificial Intelligence, Vol. II. Addison-Wesley, 1981

(王珏 袁小红)

zhijie cunchuqi cunqu

直接存储器存取 (direct memory access, DMA) 不经过中央处理器、直接用硬件实现的外围设备与主存储器之间的高速数据传送。这种传送通常利用中央处理器在执行与访问主存储器无关的操作时进行,在传送过程中不需要中央处理器介入。此时外存储器窃取了中央处理器的周期,所以 DMA 又称**周期窃取**。在周期窃取期间,程序计数器保持不变,输出输入操作与中央处理器并行工作,主存储器通过输入输出接口直接与外围设备连接。它不用程序而用专门的 DMA 控制器直接实现主存储器与外围设备之间的数据发送和接收。其传送速度

仅受存储器的数据通道宽度和外围设备的数据传送特性的限制,它是一种高速数据传输操作。常用于高速外围设备与主存储器、主存储器与主存储器、外围设备与外围设备之间的大量数据块传送操作。

在 DMA 传送数据时要占用系统总线。根据占用总线的方法不同, DMA 可分为中央处理器停止法,总线周期分时法和总线周期挪用法等。无论采用哪种方法,在 DMA 传送数据期间,中央处理器不能使用总线。因此, DMA 传送虽然提高了数据传送率,但仍占用中央处理器时间。当然与程序控制方式相比,占用的中央处理器时间缩短了很多。因此,不同的系统都提供了与该系统配套的 DMA 控制器来实现 DMA 传送的管理。

(徐良贤)

zhijue zhuyi luoji

直觉主义逻辑 (intuitionistic logic) 根据以荷兰数学家 J. Brouwer 为代表的直觉主义观点建立起来的逻辑。直觉主义者认为:逻辑仅是数学的一部分,决非数学的基础;数学应该产生于直觉,数学的证明只能用有穷方法或构造方法,所谓命题 φ 真是指有 φ 的一个使用有穷方法或构造方法的证明。因此,直觉主义者不接纳实无穷,不承认排中律。他们认为排中律仅在有穷情况适用,并不具有普遍性,因而不能随意滥用。关于逻辑联结词,直觉主义者作了如下解释:

(1) 命题 $\varphi \wedge \psi$ 真,是指有 φ 的一个证明及 ψ 的一个证明;

(2) 命题 $\varphi \vee \psi$ 真,是指有 φ 的一个证明或 ψ 的一个证明;

(3) 命题 $\varphi \supset \psi$ 真,是指有一个证明,它能把 φ 的每个证明转变为 ψ 的一个证明;

(4) \perp 无证明;

(5) $\neg \varphi$ 定义为 $\varphi \supset \perp$ 。

第一个直觉主义逻辑系统是由 A. Heyting 于 1930 年建立的,称为海丁系统。海丁系统是一个希尔伯特型系统,其命题演算系统的公理如下:

$$\varphi \supset (\varphi \wedge \varphi)$$

$$(\varphi \wedge \psi) \supset (\psi \wedge \varphi)$$

$$(\varphi \supset \psi) \supset ((\varphi \wedge \theta) \supset (\psi \wedge \theta))$$

$$((\varphi \supset \psi) \wedge (\psi \supset \theta)) \supset (\varphi \supset \theta)$$

$$\psi \supset (\varphi \supset \psi)$$

$$(\psi \wedge (\psi \supset \varphi)) \supset \varphi$$

$$\varphi \supset (\varphi \vee \psi)$$

$$(\varphi \vee \psi) \supset (\psi \vee \varphi)$$

$$((\varphi \supset \theta) \wedge (\psi \supset \theta)) \supset ((\varphi \vee \psi) \supset \theta)$$

$$\neg \varphi \supset (\varphi \supset \psi)$$

$$((\varphi \supset \psi) \wedge (\varphi \supset \neg \psi)) \supset \neg \varphi$$

古典命题演算中的许多形式定理都仍然是海丁系统中的形式定理,但也有一些古典命题演算中的形式定理,如排中律 $\varphi \vee \neg \varphi$ 不是海丁系统中的形式定理。

在海丁系统出现之后,许多人对直觉主义逻辑进行了更深入的研究,G. Gentzen 给出了直觉主义逻辑的自然演绎系统,S. A. Kripke 在 1965 年提出了一种直觉主义逻辑的语义模型,后人称为克里普克结构,直觉主义逻辑系统关于克里普克结构是完全的。

参考文献

Bell J L, et al. A Course in Mathematical Logic. North-Holland, 1977 (宋方敏)

zhiliu dianyuan

直流电源 (direct current power supply) 为计算机各部分提供电能的设备。

计算机的直流电源有**磁放大器电源**、**整流电源**、**线性电源**、**开关电源**等,最常见的是线性电源和开关电源。开关电源又可分为串联式开关电源和无工频

变压器开关电源两种。

磁放大器电源 以磁性元件作为控制和调整环节的供电设备。磁放大器常用的有扼流圈形式和快速形式等。一般磁放大器电源以快速磁放大器作为电压补偿元件,以晶体管直流放大器作为误差检测和信号放大元件来构成负反馈有差调整系统。磁放大器稳压电源的优点是可靠性高,体积小,不会发生振荡。

整流电源 利用半导体器件或其他方法直接将交流电转换成直流电的装置。自 20 世纪 20 年代起至今已经历了旋转式变流机组(电动机-直流发电机)、静止式离子整流器和半导体整流器的几个发展阶段。目前常用半导体整流器构成整流电源,它包括整流二极管和可控硅整流器(SCR)构成的整流和可控整流电源。

图 1 是常用的整流电源电路,它是开环的,当输入交流电压 U_i 变化时,输出直流电压 U_o 随着变化。用可控硅器件构成的可控硅整流电源,具有控制简单、可靠性高和输出功率容易增大的优点。整流电源在大、中、小型计算机中都有应用。

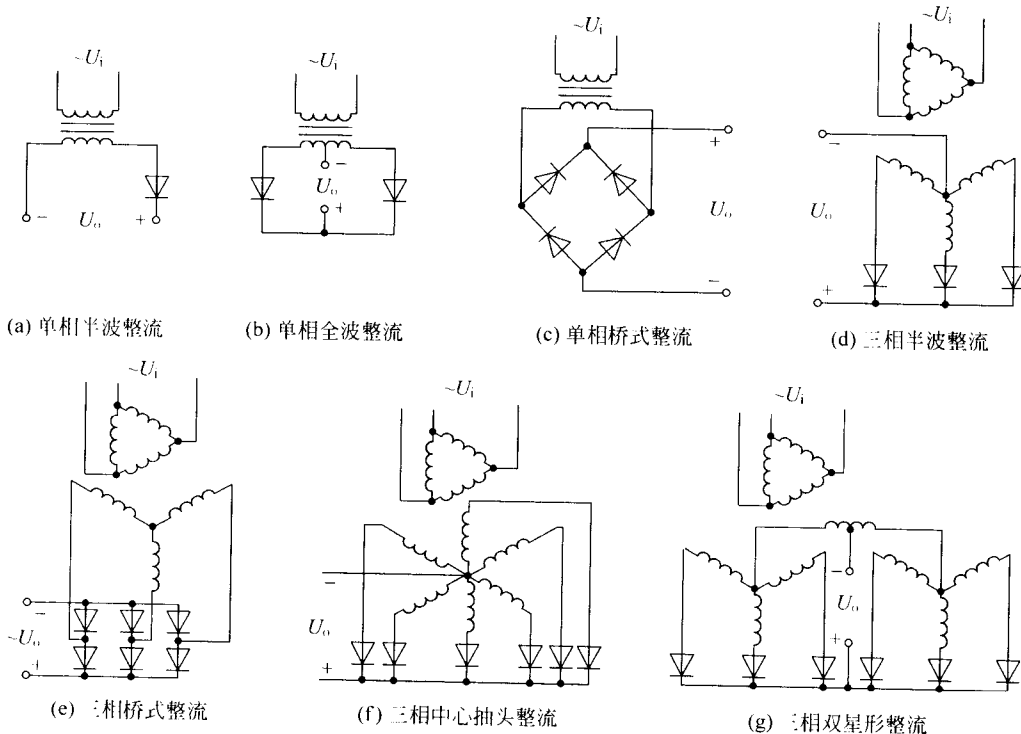
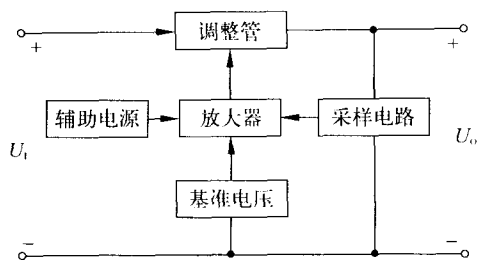
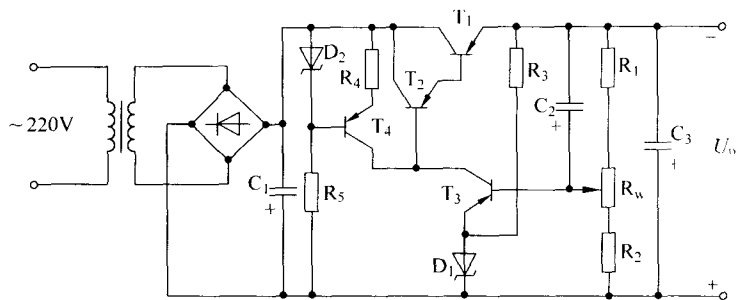


图 1 典型的单相和三相整流电源电路



(a) 框图



(b) 电路图

图2 串联式线性稳压电源

线性电源 通过调节与负荷串联的调整管压降或与负荷并联的调整管电流来达到稳定输出电压或输出电流的目的的一种供电装置。线性电源又称直流线性电源。它是一个闭环自动调整系统,包括串联方式和并联方式,其控制电路有电压控制和电流控制两种。

图2表示串联式线性稳压电源,它由调整管(通常是功率三极管或达林顿功率管)、采样电路、放大器、基准电压和辅助电源等组成。在电压控制方式的线性电源内调整管工作在功率三极管输出特性曲线的线性区,当输出滤波电容值很小时,电路的动态响应很快。输出电压的纹波也非常小,通常小于1 mV。稳定度可达1%~0.1%,噪声也很小。

线性电源应用广泛,100 W以下的低压电源已经集成化,使用非常方便。利用集成稳压芯片和运算放大器等器件组成的稳压电源线路十分简单,它的输出电压纹波小,稳压精度高,且动态响应快。

开关电源 通过调节功率器件的开通和截止时间来达到稳定输出电压和电流的供电装置。它是一个非线性闭环调节系统。自20世纪60年代起开关电源发展迅速,是计算机电源最常见的形式。后来开关电源的功率已达千瓦级。集成化模块电源的输出功率也不断提高,输入48 V、输出5 V 20 A模块等

产品也已上市。

开关电源的基本形式有4种:串联、并联、反极性以及古卡,分别见图3(a),(b),(c),(d)。这些基本形式的前级通常是工频变压器和整流滤波电路。开关电源的控制方式主要有4种:①自由振荡型,即开关频率、开通和关断时间均在一定范围内变化;②脉宽调制型,即频率固定,开通或关断时间可变;③谐振型,即开通或关断时间固定,频率可变;④移相型,电路内至少有两个开关器件,同一频率的两列方波之间的相位差可变。

无工频变压器开关电源是用高频变压器代替工频变压器,电源的尺寸和重量显著减小。无工频变压器开关电源的常用形式有正激、反激、推挽、半桥、全桥、古卡和串、并联谐振等。图4是典型的无工频变压器开关电源方框图,它包括抗电磁干扰(EMI)滤波器、桥式整流器、输入滤波器、开关晶体管、高频变压器、次级二极管整流和输出滤波器,控制部分采用线性集成电路。

微型计算机中常用的是无工频变压器开关电源。新的技术,如开关损耗为零、抑制电磁干扰、校正功率因数等,也已得到广泛应用。

三种常见的稳压电源的性能比较见表1。

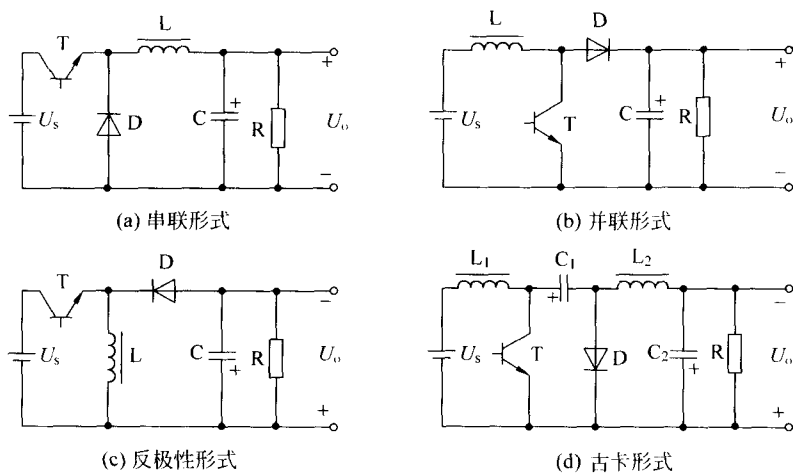


图3 开关电源基本形式

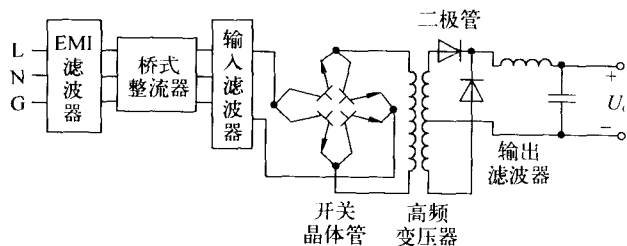


图4 典型的无工频变压器开关电源方框图

表1 三种稳压电源性能比较

比较项目 \ 电压类型	线性稳压电源	串联式开关稳压电源	无工频变压器 开关稳压电源
电压稳定度(电网变化 $\pm 10\%$)	0.01% ~ 0.05%	0.1% ~ 0.2%	0.1% ~ 0.2%
负载稳定度(负载变化0 ~ 100%)	0.02% ~ 0.05%	0.1% ~ 0.2%	0.1% ~ 0.2%
输出纹波(峰-峰值)	< 5 mV	10 ~ 100 mV	10 ~ 100 mV
上升建立时间	100 μ s	100 ms	100 ms
电压维持时间	5 ms	25 ms	25 ms
过渡过程	0.02 ~ 0.1 ms	0.5 ms 至数 ms	0.5 ms 至数 ms
效率	30% ~ 40%	40% ~ 70%	60% ~ 85%
体积	1	2/3 ~ 1/3	1/3 ~ 1/5
重量	1	2/3	1/3
可靠性(MTBF)	100 000 h	50 000 ~ 100 000 h	30 000 ~ 100 000 h
控制电路元件个数	10 ~ 30 个	10 ~ 50 个	30 ~ 120 个

计算机的发展对电源装置的性能指标,特别是电源的尺寸、重量、可靠性以及可维性等要求逐步提

高,促使电源向高功率密度和高可靠的方向发展。开关电源高频化是电源小型化的重要途径,开关电

源的工作频率为 20 ~ 50 kHz, 将提高到 100 ~ 500 kHz 以上。频率的提高不仅使电源重量减轻, 体积减小, 效率提高, 动态特性和小信号扰动下的稳定性改善, 并且使传导或辐射的电磁干扰容易被滤波、屏蔽和抑制。

(方资端 林兼)

zhi

值 (value) 可直接进行计算、储存、作为复合数据的成分, 传送作为函数或过程的变元, 以及作为函数结果所回送的任何实体。或称为计算中存在的任何实体。

值是计算机科学中最为基本的概念之一。语言不同, 值的种类也各异, 例如, 在 PASCAL 语言中可区分如下几类值: 初等值 (逻辑值、字符、枚举值、整数、实数), 复合值 (记录、数组、集合、文件), 指引元, 变量引用, 过程及函数抽象; 而在 ML 语言中却有如下几类值: 初等值 (逻辑值、整数、实数、串), 复合值 (元组、记录、构造、表、数组), 函数抽象, 变量引用。

参考文献

Watt D A. Programming Language Concepts and Paradigms. Prentice Hall, 1990

(徐家福)

zhidu cunchuqi

只读存储器 (read only memory, ROM) 一次写入信息后能长期存储, 此后只可读取的一种存储器。在一般情况下, ROM 所存的内容在较长时间内是相对固定的, 即使在断电情况下, 所存信息仍不会消失, 因此 ROM 是非易失性存储器。

早期的只读存储器采用分立元器件, 如二极管矩阵、磁心矩阵、变压器和电容器等, 后来用半导体集成电路技术制造只读存储器。半导体只读存储器的基本存储模块为只读存储器电路芯片。电路主体是一个由大量存储单元组成的存储阵列, 每个存储单元主要由位于阵列字线和位线交叉处的半导体管构成, 半导体管通或断分别代表存 1 或存 0。半导体只读存储器写入数据的过程称为对 ROM 编程。根据编程方式的不同, 半导体 ROM 可分为三类:

①掩模型只读存储器 (掩模 ROM)。这类 ROM 所存的数据在芯片制造过程中完成编程。使用时只能读出, 不能写入。②可编程只读存储器 (PROM)。这类存储器所存的数据可由用户根据自己的需求进行一次性编程。此后, 存储器只能读出, 不能写入。

③可改写只读存储器。这类存储器有可擦编程只读存储器 (EPROM) 和电可修改只读存储器 (EAROM) 两种。这类存储器可用紫外光照射或电的方法擦除已写入的数据, 然后再用电的方法重新写入新的数据。用户可以根据自己的需要多次改写 ROM 内容。由于 EPROM 和 EAROM 改写时间比读出时间长得多, 有些电路的改写又需在专门的电路上进行, 故它们在工作时只读不写, 仍属于只读存储器。

只读存储器广泛用作各种类型的固件, 如用于存放接通电源后启动计算机的引导程序、高级语言的加工程序和各种微程序。只读存储器经常被做成代码变换器、字符产生器、数学函数表、汉字字型库、游戏卡、音乐卡等, 也可实现各种组合电路功能, 配套构成时序电路, 还可以 ROM 网络的形式实现特殊的逻辑函数需求。由于只读存储器能有效地提高计算机软硬件的功能并且具有可靠性、灵活性及经济性的特点, 所以它已成为现代计算机系统的重要部件。

除了半导体只读存储器外, 还有一类只读存储器, 它们按直接存取方式工作, 这就是光盘存储器中的只读光盘存储器 (CD-ROM) 和一写多读光盘存储器。只读光盘存储器上的信息是母盘经模压复制在光盘上的, 只具有读取信息的功能, 用户不能改写。只读光盘是由视听光盘演变而来, 用在计算机上的 CD-ROM 与 VCD、SVCD、DVD 等家电产品的光盘结构相同, 但记录格式不同。CD-ROM 可用以存储各类软件产品, 如系统软件、应用软件、专家系统、多媒体信息以及各种电子出版物等。由于光盘容易复制、价格便宜、容易使用, 已非常流行。一写多读光盘存储器的写入过程是激光束使存储介质相应部位“烧蚀”, 已经永久性地改变了其光学性质, 写后不能修改, 但可多次读取。它常用以存储文件、档案、资料等。

参考文献

1. 张江陵, 金海. 信息存储技术原理. 武汉: 华中理工大学出版社, 2000

2. Hodges D A. Semiconductor Memories. IEEE Press, 1972

(时万春)

zhidu cunchuqi xin pian

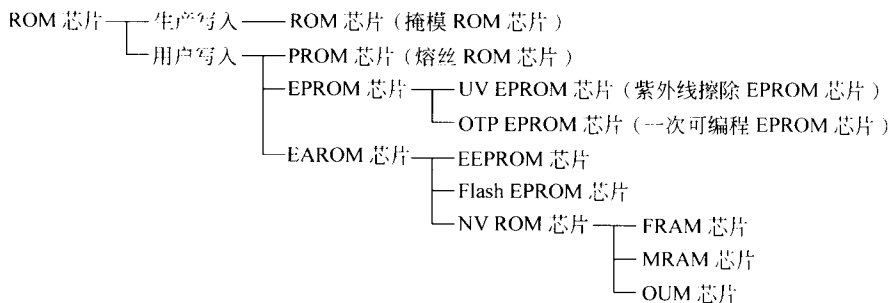
只读存储器芯片 (read only memory chip) 在正常运行情况下, 只有读出操作, 没有写入操作的半导体存储器芯片。简称 ROM 芯片。写入数据后, 即使切断电源, ROM 芯片存储内容仍不会消失, 所以是非易失性存储器。随着微电子和存储器技术的发

展,习惯上把具有系统运行中写入数据能力,但应用中以读出操作为主的非易失性存储器也视为 ROM 芯片。

ROM 芯片分类 ROM 芯片产品是 1970 年问世的,早期产品是掩模型只读存储器芯片(mask ROM 芯片),其存储内容是在集成电路生产过程中由集成电路生产者按用户要求写入的。随后出现了可由用户自行编程,但只能一次性写入数据的 ROM 芯片,即基于双极工艺的高速熔丝型可编程只读存储器芯片(PROM 芯片)。1971 年又生产出可由用户编程写入数据并且可擦除的可擦编程只读存储器

芯片(EPROM 芯片),它是一种采用 FAMOS 工艺制造的紫外线擦除的 EPROM 芯片,记为 UV EPROM 芯片。随着系统运行中可编程需求的增长,1977 年研制出可现场修改存储内容的电可修改只读存储器芯片(EAROM 芯片),称之为电可擦编程只读存储器芯片(EEPROM 芯片)。20 世纪 80 年代以来陆续推出一些新的 ROM 芯片型式,如一次可编程 EPROM 芯片(OTP EPROM 芯片)、快可擦编程只读存储器芯片(Flash EPROM 芯片)和非易失性随机存取存储器芯片(NV RAM 芯片)等。表 1 为只读存储器芯片分类。

表 1 ROM 芯片分类



掩模型只读存储器芯片 早期的掩模 ROM 芯片以 NMOS 管作为基本存储单元,后来过渡到以 CMOS 为主。图 1(a) 为一个 4 字 × 4 位掩模 ROM 阵列,阵列中连接 MOS 管的数据为 1,反之为 0。图中存储的 4 个字分别为 1010,0100,0110 和 1000。这种 MOS 结构从字线方向看,是一个 4 输入端“或非”门,通常称为 NOR 阵列。读出操作时,经地址译码后在被选的字线上加高电平,非选字线为低电平。高电平字线使选中的 MOS 管通导,相应位线输出低电平,没有选中的 MOS 管的位线输出高电平。位线输出经反相后得到 ROM 存储的数据代码。除 NOR 阵列外,还有 NAND 阵列,如图 1(b)所示。

早期的掩模 ROM 主要用于大型计算机操作系统存储媒体,20 世纪 80 年代中期电子游戏机的发展,特别是它对大容量、高集成度、低成本 ROM 的需求促成了掩模 ROM 市场的发展,并且长盛不衰。据 1988 年统计,掩模 ROM 市场的 54% 用于消费性产品,其中游戏机占 46%,电子乐器占 8%;而数据处理市场仅占 43%,主要用于文字处理机(19%)、个人计算机和 workstation(10%)、办公室自动化(5%)、打印机等(9%);工业应用市场很小,不及 3%。图 2 为 1Mb CMOS 掩模 ROM 芯片框图,与早期低集成度

(如 1 kb ROM)产品相比,主要是增加了输入地址和输出锁存功能。

可编程只读存储器芯片 一种可按用户要求,由用户一次性写入内容的只读存储器芯片(参见可编程只读存储器芯片)。

可擦编程只读存储器芯片 一种可改写的只读存储器芯片,存储内容可由用户写入,芯片从电路系统中取出后,用紫外线照射可擦除存储内容(参见可擦编程只读存储器芯片)。

电可擦编程只读存储器芯片 一种可改写的只读存储器芯片,其存储内容可由用户通过电信号进行写入和擦除(参见电可擦编程只读存储器芯片)。

快可擦编程只读存储器芯片 一种可改写的只读存储器芯片,其存储内容可由用户通过电信号进行写入和擦除。与电可擦编程只读存储器芯片不同的是,其存储单元是单管结构(参见快可擦编程只读存储器芯片)。

非易失性随机存取存储器芯片 NV RAM 芯片是 ROM 芯片族中一个新系列,它将标准的 RAM 阵列和 EEPROM 阵列组合在同一芯片上。当芯片加电时,EEPROM 内容自动写入 RAM,提供原先保存的系统配置参数。系统运行中上述参数可通过 RAM

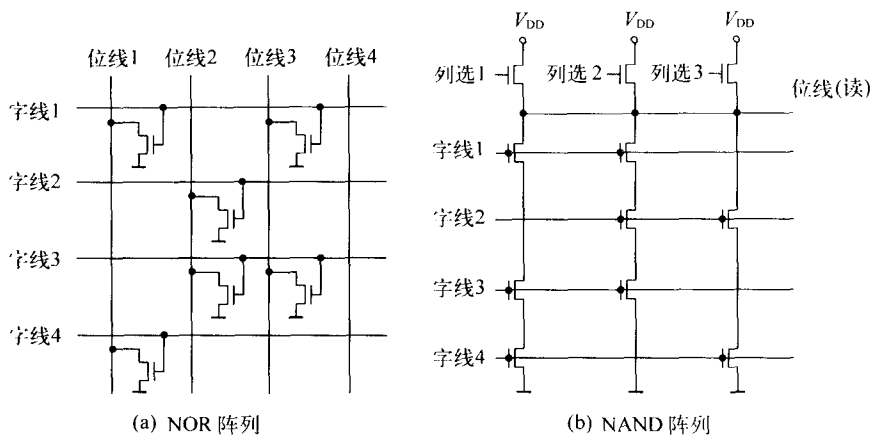


图1 掩模 ROM 阵列

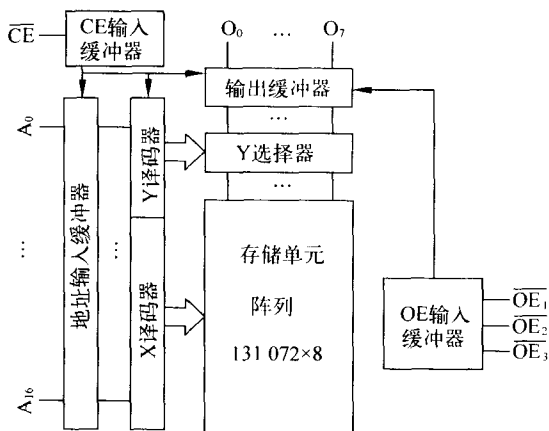


图2 1 Mb(128kb×8)掩模 ROM 芯片框图

进行修改,并通过芯片内 RAM 和 EEPROM 之间的数据传送实现 EEPROM 芯片的重新编程,以实现系统配置参数的修改。NV RAM 芯片有两种型式:基于静态 RAM 单元阵列的 NV SRAM 芯片和基于动态 RAM 存储阵列的 NV DRAM 芯片。NV RAM 芯片具有随机存取和非易失的性能,适合于存储容量不大,存储参数不多,需高速存取并且仅仅偶然修改存储内容的场合。目前主要用于系统加电或断电时恢复和保存系统设定值,还广泛用于办公室可视终端、销售点终端、工业控制和机器人等。

NV RAM 芯片源于装有后备电池的 SRAM 芯片,即 B SRAM 组件。B SRAM 组件内存放一些锂电池,当外部电源切断时,组件内的电压监测电路控制将其切换到由锂电池供电,以保护 SRAM 内容,不

使丢失。第一个 NV SRAM 芯片是 1977 年研制的,取其结构特征而称之为影像 SRAM。原型是一个 256 b 的 SRAM 芯片阵列,其上面像影子一样一一对应地覆盖 256 b EEPROM。图 3 为 1988 年开发的 64 kb NV SRAM 芯片产品的功能框图。图中的存储阵列调用引脚表示了它独具的特性,可控制 SRAM 存储阵列和非易失 EEPROM 存储阵列之间的数据传送。该芯片内包含一个电荷泵,所以只需 5 V 供电。NV DRAM 芯片开发于 20 世纪 80 年代末,它保持了 DRAM 芯片的单管加一个存储电容的单元结构,只是该电容具有 EEPROM 芯片单元电容的非易失性能。

ROM 芯片的发展 随着人们对更高密度、更大带宽、更低功耗、更短延迟时间、更低成本和更高

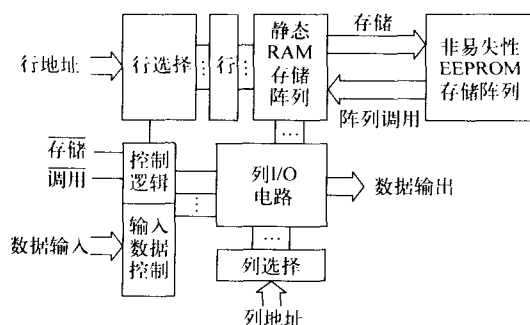


图3 64 kb NV SRAM 芯片功能框图

可靠性存储器芯片的追求和各种便携式产品的发展,在终端产品轻、薄、短、小的要求下,一些新型 NV RAM 芯片相继出现。比较有竞争力和发展潜力的 NV RAM 芯片有铁电随机存取存储器(FRAM)芯片、磁电阻式随机存取存储器(MRAM)芯片和相变存储器(OUM)芯片。

FRAM 芯片是一种在断电时不会丢失内容的非易失性存储器芯片,具有高速、高密度、低功耗和抗辐射等优点。FRAM 芯片的存储原理是基于铁电材料的高介电常数和铁电极化特性。按工作模式可分为破坏性读出和非破坏性读出。已开发完成的 32 Mb FRAM 芯片采用单管单电容的单元结构和 $0.2\ \mu\text{m}$ 工艺制造,存取时间为 50 ns,循环周期为 75 ns,工作电压为 3 V。但后来 FRAM 芯片在密度方面的发展不甚顺利。

MRAM 芯片通过控制铁磁体中电子旋转方向来改变读取电流大小,从而使其具备二进制数据存储能力。理论上铁磁体是永远不会失效的,因此它的写入次数也是无限的。自从 2002 年 6 月研制出 1 Mb MRAM 芯片,2003 年推出了 $0.18\ \mu\text{m}$ 工艺的 4 Mb MRAM 芯片样片以来,MRAM 芯片发展迅速,有望逐步取代现有的存储器而得以广泛应用。MRAM 芯片面临的主要技术挑战是磁致电阻太过微弱,两个状态之间的电阻差异只有 30%~40%,读写过程中要准确识别差异,有一定的难度。

OUM 芯片是一种基于 Ovshinsky 电效应的存储器芯片。其原理是利用 Gb、Sb、Te 等为材质的薄膜来存储数据,数据存储方式类似 CD-ROM,利用温度造成的相位变化存储数据。OUM 芯片的优点是产品体积小、成本低,可直接复写,易于编程,也就是在写入数据时不用将原有数据擦除。早在 2001 年 7 月就有 $0.18\ \mu\text{m}$ 工艺的 4 Mb OUM 测试芯片问世。

OUM 芯片发展中的技术重点是如何稳定维持其驱动温度。

半导体只读存储器芯片尽管灵活性差,但因具有高集成和低成本的特点,得以持续地占有非易失存储市场的很大份额。在半导体只读存储器芯片市场中,掩模 ROM 芯片将继续占领一定的市场份额;现场可修改 ROM 芯片中,Flash EPROM 芯片和 NV RAM 芯片(特别是其中的 NV DRAM 芯片)由于功能特性好、成本低,将获得较快发展。

参考文献

1. Prince B. Semiconductor Memories. 2nd ed. Chichester: John Wiley & Sons, 1991
2. T P Ma, J P Han. Why is Nonvolatile Ferroelectric Memory Field-Effect Transistor still Elusive? IEEE Electron Device letters, 2002, 23: 386~388
3. Lind e Geppert. The New Indelible Memories. IEEE Spectrum, 2003, 40(3): 48~54 (时万春)

zhidu guangpan qudongqi

只读光盘驱动器 (read only optical disc drive) 控制并读出只读光盘(CD-ROM)上所存信息内容的设备。又称只读光盘机。CD-ROM 中能存储大量信息,获得高质量图像和高保真音乐,现在广泛应用于文献数据库、多媒体信息存储及计算机辅助教育等方面。

CD-ROM 是一种光盘。现在普遍采用的格式标准是 ISO 于 1988 年制定的 ISO 9 660。标准规定 CD-ROM 的直径是 120 mm (4.75 in),厚 1.2 mm。该光盘用折射率为 1.5 的透明材料(常用的是聚碳酸酯)制成。在存储媒体表面用压印工艺形成一连串的“凹坑”和“平台”,凹坑和平台在存储表面上组成一个螺旋线状的光轨。光轨的间距为 $1.6\ \mu\text{m}$,相当于每英寸 16 000 条光轨。凹坑的宽度约 $0.11\ \mu\text{m}$,深度 $0.5\ \mu\text{m}$,长度则由存储的内容确定,一般在 $0.833\sim 3.05\ \mu\text{m}$ 之间。每片盘只使用一面,可存储容量为 680 MB。在记录立体声音乐时,可放音最长时间是 74 min。

CD-ROM 可存储数字化的图形、图像、音乐、文字以及数字等内容。标准规定存储以“扇区”为单位,每个扇区头上有地址区,占 28 个字节;中间是用户数据区,占 2 048 个字节;尾部是纠检错码区,占 288 个字节。也可以不要纠检错码,数据区扩大到 2 336 个字节。

CD-ROM 采用 8/14 调制码(EFM)(参见光记

录编码方法), 纠检错用交叉 R-S 码(CIRC)(参见数字磁记录纠错码)。

CD-ROM 的存储内容是用压印工艺制成的, 用户不能更改, 因此只读光盘机没有写入功能, 只有读出、寻道等功能。在结构上可分成: 读出光盘头, 寻道跟踪机构和电路, 主轴稳速机构和电路, 自动聚焦机构和电路, 读出和纠错电路以及接口电路等部分。

光盘头由激光发生器和接收器组成。激光发生器现在常用半导体发生器, 波长为 780 nm ~ 790 nm (近红外光)。在自动聚焦机构和寻道机构的控制下, 激光被聚焦对准在光轨上。当激光束照射到平台时, 有反射光照到用光敏元件组成的接收器, 当射到凹坑时, 反射光非常微弱。利用反射光的强弱, 接收器检测出所存储的内容是 1 还是 0。在实际中并不采用凹坑表示 1, 平台表示 0 的方法, 而是采用磁记录中常用的逢 1 变化不归零制(NRZI)编码方法, 用凹坑的上升边和下降边表示 1。读出的光信号经过解码和纠错, 得到所存储的信息。在读出过程中, 光盘头与光盘表面不接触。

CD-ROM 采用等密度记录, 因此主轴要保持等线速度(CLV)。利用读出信号中的同步信号进行控制, 可保证线速度为 1.2 m/s。相应的数据传输速率是 150 kB/s。这一传输速率适用于音乐, 但不能满足图像、视频的要求。为了提高数据传输速率, 现在已有两倍速、四倍速的只读光盘机, 把数据传输速率提高到 300 kB/s 和 600 kB/s。

只读光盘机与计算机间多采用 SCSI 接口, 现在也有采用 IDE 接口的。

除了上述标准的 CD-ROM 盘外, 现在还有小尺寸的 CD-ROM 盘。如 CD-3 (80 mm 盘径) 盘、Mini Disk 盘(64 mm 盘径)。这些 CD-ROM 盘在记录格式上和 120 mm 标准盘兼容, 适用于便携式计算机。

CD-ROM 盘在物理结构上和记录方法上与激光唱盘(CD-DA)是相同的, 只是记录格式不同, 在只读光盘机上可以利用软件控制来使用激光唱盘, 通过扬声器放出音乐。

ISO 9660 标准规定 CD-ROM 盘可以存储图像、文字和声音, 但不能同时读出两种形式的数据。当需要画面配合有声音时, 要先读出其中一种, 存入计算机, 再读另一种, 然后由计算机处理合成, 使用很不方便。1988 年对标准进行了一次修订, 扩展成 CD-ROM/XA 标准, 把文字、声音、图像信息交错地存放在盘面上, 可获得较好的多媒体效果。Kodak 公司还提出 Photo-CD 标准, 主要用来存储照片。还

有采用 CD-ROM 标准制造的一次写入多次读出的 CD-R 盘。目前许多型号的只读光盘机对这些标准都有兼容性, 因而对此类只读光盘都能读出。

CD-ROM 盘工艺简单, 适用于大批量生产。光盘机采用不接触式读出, 对灰尘不像磁盘那样敏感, 且抗磁干扰, 存储量大。因此广泛用作各种电子出版物, 如百科全书、年鉴、手册、技术说明书、软件、车载地图、动画、电子游戏等。只读光盘机已成为个人计算机特别是多媒体计算机的一种标准配置设备。

参考文献

Pohlmann K C. The Compact Disc: A Handbook of Theory and Use. Oxford: Oxford University Press, 1993 (余胜生)

zhicheng yuyi

指称语义 (denotational semantics) 把语言成分映射为数学对象, 然后用定义在对象上的运算所表达出的语言的语义。

指称语义的主要思想是英国牛津大学 C. Strachey 于 1964 年前后提出的。美国 D. Scott 创建的论域理论为指称语义学奠定了数学基础。论域理论是讨论各种语言成分指称物的数学结构, 以及提供在这种数学结构上定义语言成分的语义和推导语言成分特性所必须的数学工具。

IBM 公司维也纳实验室在研究操作语义方法 VDL 的基础上, 于 70 年代初转向指称语义的研究, 发展了基于指称语义方法的一整套开发软件的工程方法, 称为维也纳开发方法(简称 VDM)。

1976 年英国一些学者发展了论域理论, 提出幂论域理论, 从而为定义非确定性程序的指称语义奠定了理论基础。指称语义方法定义语言的语义基本思想是: 先确定指称物, 然后给出语言成分至指称物的语义映象。这个映象要求满足:

(1) 每个成分都对应有指称物;

(2) 复合成分的指称只依赖于它的子成分的指称。

下面以算术表达式和赋值语句及顺序语句为例给以简要说明。

算术表达式的效果就是根据程序变量当前值计算表达式的值。程序变量的当前值可以用一个数值向量来表示。如果有 k 个程序变量 x_1, x_2, \dots, x_k , 则 k 维数值向量 (n_1, n_2, \dots, n_k) 表示 x_1 的值为 n_1, x_2 的值为 n_2, \dots, x_k 的值为 n_k 。程序变量的一种取值称为程序的一个状态。状态的全体集合称为状态空

间,记作 $State$ 。算术表达式的值的范围记作 Num 。算术表达式的指称物就是 $State$ 至 Num 的一个映象,也就是根据 $State$ 中的任意一个元素(即程序变量的一组取值)可求得 Num 中对应的一个元素(即表达式在变量的这组取值下的值)。数学中的映象只反映集合间元素的对应,用映象作为语言成分的指称物在语义的定义中就避免了涉及语言成分的执行过程。 $State$ 至 Num 的全体映象,即全体表达式的指称物,记作 $State \rightarrow Num$ 。不同表达式的指称物是不同的,对算术表达式的语义下定义,就是要对每个算术表达式至其指称物的一个对应下定义,故也可表示为一种映象。用 Exp 表示算术表达式的全体,那么 Exp 的语义,就是 Exp 至指称物集合($State$, Num)的映象,记作

$$Exp \rightarrow (State \rightarrow Num)$$

常量 n 是一种算术表达式,变量 x 本身也是一种算术表达式,两个算术表达式的和及积等也是算术表达式,故算术表达式 Exp 的抽象语法可用巴科斯范式(BNF)表示:

$$E ::= n \mid x \mid e + e \mid e \times e \mid \dots$$

指称语义映象 D 可定义如下:

$$D[n](S) = n$$

$$D[x_i](S) = S_i$$

$$D[e_1 + e_2](S) = D[e_1](S) + D[e_2](S)$$

$$D[e_1 \times e_2](S) = D[e_1](S) \times D[e_2](S)$$

第一个公式表示,无论程序处于何种状态 S ,常量 n 的值不变,为数学中相应的量 n 。为了区分元语言和程序设计语言,指称语义定义中将程序设计语言中的成分用 $|$ 括起来。

第二个公式表示,变量 x_i 在状态 S 时的值为数值向量 S 的第 i 个分量 S_i 。

第三、四个公式表示,表达式 $(e_1 + e_2)$ 和 $e_1 \times e_2$ 在状态 S 时的值分别为子表达式 e_1 和 e_2 在状态 S 时的值的和与积。

表达式的抽象语法规则如何用最简单的表达式常量和变量构成其他表达式。而表达式的语义定义也是先给出最简单的表达式的语义,然后按照语法的构造过程去定义其他表达式的语义,使得复合成分的语义由各成分语义复合而成。这种定义语义的方法叫作结构式方法,或叫语法引导方法。指称语义学就是一种结构式形式语义学。执行程序设计中语言中的语句导致程序状态的变化(即程序变量取值的改变),故语句的指称物应是 $State$ 至 $State$ 的映象: $State \rightarrow State$ 。定义语句的语义就相当于定义映

象 D :

$$D: Sts \rightarrow (State \rightarrow State)$$

Sts 表示所有语句的集合。如表示赋值语句 $(x_i := e)$ 的语义即改变状态 S 为 $S \left| \begin{smallmatrix} D[e](S) \\ i \end{smallmatrix} \right.$ 。

$S \left| \begin{smallmatrix} D[e](S) \\ i \end{smallmatrix} \right.$ 表示将 S 中的第 i 个分量(即 x_i 的值)改变为表达式 e 在状态 S 时所取的值(即 $D[e](S)$)。

$$D[S_1; S_2](S) = D[S_2](D[S_1](S))$$

这表示顺序执行语句 S_1 和 S_2 ,即先执行 S_1 ,并将执行 S_1 后形成的新状态,作为当前状态再执行 S_2 。

实际的程序设计语言非常复杂,所定义的语义映象比之上文列举的更为复杂。为了允许同名变量在不同过程体中表示的值不同,指称语义中引进环境的概念;为了刻画程序控制的转移又引进后续的概念,这些概念在描述和简化语义定义中有重要的作用。

参考文献

周巢尘. 形式语义学引论. 长沙: 湖南科技出版社, 1985 (周巢尘 李晓山)

zhihui kongzhi tongxin qingbao xitong

指挥控制通信情报系统 (command, control, communication and intelligence system) 参见军事指挥信息系统。

zhiling geshi

指令格式 (instruction format) 计算机指令的表示形式。

一条指令通常包含下列信息:

(1) 操作码 说明指令所执行的操作,例如加法、移位或转移等。一台计算机可能有几十条甚至几百条指令,每一条指令都有一个相应的操作码。操作码的长度与机器所含的指令条数有关。

(2) 操作数来源 在指令中指出操作数的地址(寄存器地址或存储器地址),根据地址取得操作数。

(3) 目的地址 运算结果保存在目的地址中,可以是寄存器地址或存储器地址。

(4) 下一条指令的地址 该地址由程序计数器提供。当程序顺序执行时,每执行一条指令后,程序计数器的内容加1,得到下一条指令的地址,而当执行转移指令(或其他改变执行顺序的指令)时,由本

条指令指出下一条指令的地址。

指令由操作码和地址码组成,地址码包括上述(2),(3),(4)的内容。根据地址码部分所包含的地址个数来划分指令,可以有零地址、一地址、二地址和三地址等多种指令格式,如图1所示。其中OP为操作码, A_1, A_2, A_3 为地址码。

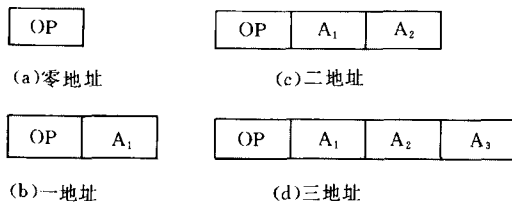


图1 指令格式

零地址指令只有操作码,没有地址码,适用于两种情况:①空操作、停机等不需要地址码的指令;②操作数地址是隐含的,如堆栈指令,由堆栈指针给出地址。

一个地址指令在指令中只给出1个地址,例如“加1”、“减1”等单操作数指令就采用这种格式。又如某些计算机的双操作数指令,已约定一个操作数在某个指定的寄存器中,指令中只需要指出另一个操作数地址,运算结果通常也是送回到这个约定的寄存器中。

二地址指令是广泛被采用的双操作数指令, A_1, A_2 是两个操作数的地址,其中一个(A_1 或 A_2)还是目的地址,算术逻辑运算指令一般都是二地址指令。

某些计算机采用三地址指令,其中两个地址为两个操作数的地址,第三个为目的地址。

以上所述是经常见到的几种指令格式,一台计算机可能只具有其中一部分指令格式。寄存器地址的长度只需要几位就够了,存储器地址的长度取决于寻址方式和存储器容量,一般在20位到32位之间。

精简指令集计算机的指令长度一般是固定的,操作码和地址码的长度及其在指令中的位置一般也是固定的。复杂指令集计算机的指令长度一般不是固定的,它随指令而异,有的计算机的指令长度甚至在一个字节到十几个字节之间变化,操作码的长度也不是固定的。应尽量把常用指令的长度设计得短一些,以节省存储空间和减少访问存储器的次数。

(王爱英)

zhilingji bingxing chuli

指令级并行处理 (instruction level parallel processing, ILPP) 在处理机的指令级上提高指令执行并行性的软件和硬件相结合的技术。

并行处理一般是指多处理机系统或多计算机系统所进行的信息处理,这往往是在任务级或在进程级上的并行处理,其目标是提高整个多处理机系统或整个多计算机系统的性能。指令级并行处理是在精简指令集计算机(RISC)问世以后发展起来的,其目标是提高处理机本身的处理速度,而对于应用软件来说是完全透明的。

先进的处理机体系结构都采用流水线技术(参见计算机流水线),使多条指令的执行重叠进行,以提高处理机的运行速度。流水线执行中存在转移相关和数据相关问题。为了解决这些问题,在软件方面,采用编译优化,尤其着重其中的指令重组;在硬件方面,有解决数据相关的相关检测电路和内部提前数据通路等技术以及解决转移相关的转移预测技术等。

在多发射结构中,每个机器周期发射多条指令,不仅同一个机器周期内的多条指令之间存在着相关,而且邻近周期的各条指令之间也存在着同样的问题。因此,解决流水线中的相关性问题的困难。

解决流水线中相关性问题的可以采用硬件和软件相结合的指令调度技术。通过指令调度以解除或减少程序中的指令间的相关,使更多的指令可以并行执行而保持流水线畅通,从而提高处理机执行指令的速度。

衡量指令执行并行性的参数为IPL:

$$IPL = \frac{\text{程序执行总指令数}}{\text{程序执行总周期数}}$$

进行指令调度的目的是提高IPL值。

指令调度应在一个确定的范围内进行,通常称这个范围为一个基本块。基本块的定义是:它是只有1个入口点和1个出口点的代码段。该代码段中没有转移或跳转等分支指令。段头为入口点,段尾为出口点。在一个基本块内进行指令调度,避免了转移相关问题,只需解决数据相关和资源冲突问题。但是,经过统计分析,一般程序中的基本块平均包含的指令数只有6条~8条。这使指令调度的作用有限,IPL一般不超过2。

要进一步提高IPL,必须考虑跨基本块之间的指令调度,或者扩大基本块。另外,循环在程序中是

频繁出现的,如果把循环优化并使各个循环中的指令得以重叠执行,也可使 IPL 提高较多。这些都是指令级并行处理的内容。

基本块以内的指令调度称为局部调度;跨基本块的指令调度称为全局调度。从程序执行角度,调度可以分成静态调度和动态调度。静态调度是先把源程序进行编译,生成目标码,再进行调度,然后执行;动态调度是在执行目标码的过程中进行调度。前者主要靠软件方法实现,后者则用软件和硬件相结合的方法实现。

从硬件实现调度的角度,也可有两种级别上的调度:一是在计算机组织级别上,如转移预测缓冲、寄存器记分牌和内部提前技术;二是在计算机体系结构级别上,如多发射结构。

基本块以内的指令调度方法有寄存器分配法、指令重新排序法、指令压缩法等,其中所使用的算法有线性压缩算法、搜索算法、列表调度算法和关键路径算法等。

跨基本块的指令调度方法有静态转移预测法、动态转移预测法、踪迹调度法和冒险执行法等。

指令级并行处理对处理机体系结构和编译优化技术有重大影响,对提高处理机的性能起关键性作用。

参考文献

李三立,李亚民. RISC——单发射与多发射体系结构. 北京:清华大学出版社,1994 (李三立)

zhiling jicunqi

指令寄存器(instruction register) 用来存放当前正在执行的指令的寄存器。计算机的所有操作都是通过分析存放在指令寄存器中的指令后再执行的。当指令从主存储器取出后,将其暂放在指令寄存器中,以便在指令执行过程中,完成一条指令的全部功能控制。指令内容包含有确定操作类型的操作码和指出操作数来源或去向的地址。操作码部分送到译码电路进行分析,指出本指令将执行何种类型的操作;地址部分送到地址形成部件生成有效地址,作为取数或存数的地址。(刘恩德)

zhiling leixing

指令类型(instruction type) 根据指令完成的主要功能而对指令所进行的分类。

一台计算机通常有几十条到几百条指令,按其

所完成的功能可分为:算术逻辑运算指令、移位指令、浮点运算指令、十进制运算指令、数据传送指令、转移指令、字符串处理指令、向量运算指令、堆栈指令、输入输出指令、特权指令和控制指令等。但并不要求一台计算机具有上述所有类型的指令。

(1) 算术逻辑运算指令

计算机一般都具有这类指令。早期的小型计算机和微型计算机的硬件结构比较简单,一般只设置二进制定点加法、比较和求补码(取负数)等最基本的算术指令。由于芯片集成度的提高,后来的中央处理器都支持用硬件实现的乘除法指令。计算机还具有对两个操作数进行逻辑乘、逻辑加和按位加(异或)等操作的逻辑运算。有些计算机还设置位操作指令,如位测试(测试指定位的值)、位清除(把数中的某一位置为零)、位求反(取某位的非值)等指令。

(2) 移位指令

分算术移位、逻辑移位和循环移位3种。可以对 n 位操作数左移或右移 $0 \sim n-1$ 位。算术移位和逻辑移位的主要差别在于右移时,填入高位的数码不同,算术移位处理的是带符号的操作数,所以右移时保持最高位(符号位)不变;逻辑移位处理的是不带符号的操作数,右移时,最高位填入0。算术左移和逻辑左移的操作是相同的,低位补充0。循环左移将移出的最高位送到最低位;循环右移将移出的最低位送到最高位,使数据本身循环传送。

移位还可实现对带符号数或不带符号数乘以 2^i 或整除以 2^i 的运算(分别左移 i 位或右移 i 位)。移位指令的执行时间比乘除法运算的执行时间短得多。

(3) 浮点运算指令

浮点运算适合对数值范围变化较大的数,用于科学计算或工程计算的计算机往往设置浮点运算指令,处理的数经常为单精度(32位)或双精度(64位)数。某些计算机没有设置浮点运算指令而用子程序实现浮点运算,因而处理速度慢。

(4) 十进制运算指令

在某些数据处理系统中,输入输出数据很多,设置十进制运算指令可减少十进制数与二进制数之间的转换,提高处理速度。在通用大型计算机系统中,这些数由若干位十进制数码组成,每个十进制位用BCD码表示;在某些微处理器中设置的十进制加减法运算指令往往只对1位十进制数进行运算,对于多位的十进制数的运算则用程序实现。

(5) 数据传送指令

这类指令实现寄存器与寄存器、寄存器与存储器单元、存储器单元与存储器单元之间的数据传送或数据交换。某些计算机的 I/O 设备与存储器单元统一编址,此时的数据传送指令还包括寄存器(或存储器单元)与 I/O 设备之间的数据传送。

数据传送指令 1 次可传送 1 个数据或 1 批数据。传送 1 批数据时,执行时间较长,在传送过程中如果有中断请求,往往允许暂时中止现行指令的执行而转入中断处理程序,处理完毕返回原程序后,再从该指令中止处继续执行下去。

(6) 转移指令

这类指令用于控制程序流的转移。在大多数情况下,计算机是顺序执行程序,但也会遇到转移到另一段程序或循环执行某段程序的情况。

按转移的性质,转移指令分为无条件转移、条件转移、过程调用与返回、软中断等几种。

无条件转移指令 不受任何条件约束,将程序转移到本指令所规定的下一条指令继续执行。

条件转移指令 根据指令所指定的条件进行测试,根据测试结果的“真”、“假”来决定转移是否执行。通常利用算术指令建立的条件码来控制程序流,实现程序的分支。

调用指令和返回指令 在编写程序过程中,常常预先编写一些经常使用的,能够独立完成某一指定功能的程序段,在需要时能随时调用,而不必多次重复编写,以节省存储器空间和简化程序设计,这些程序段称为子程序或过程。操作系统也提供大量通用子程序,如申请资源、读写文件、控制外部设备等,需要时可直接调用。计算机使用调用指令 Call(过程调用、系统调用、转子程序)来实现从一个程序转移到另一程序段的操作,执行完被调用的程序段后再返回到原调用程序,继续执行,这一功能由返回指令 Return 实现。返回地址一般保留在堆栈中,随同保留在堆栈中的还有一些状态寄存器和通用寄存器的内容。

软中断(或称陷阱)和软中断指令 在计算机运行过程中,有时可能出现电源电压不稳定、存储器校验出错、输入输出设备出现故障、用户使用了未定义的指令或特权指令等意外情况,使计算机不能正常工作,如果不及时处理,会影响系统正常运行。此时,计算机发出软中断信号,并暂停当前程序,转入故障处理程序。

在某些计算机中设置有可供用户使用的软中断

指令,利用它来实现系统调用和程序请求。

(7) 字符串处理指令

用于信息管理、数据处理、办公室自动化等领域的计算机需要有很强的非数值处理能力。字符串处理指令是一种非数值处理指令。它一般包括字符串传送、字符串比较、字符串查询、字符串转换等指令。字符串传送指令可将数据块从主存的某区域传送到另一区域;字符串比较指令将一个字符串与另一字符串逐个字符进行比较,以确定是否相等;字符串查询指令可查找在字符串中是否含有某一指定的字符或子字符串;字符串转换指令可将字符串从一种表达形式转换到另一种表达形式,例如从 ASCII 码转换成 EBCDIC 码(扩充的二-十进制交换码)。

这类指令可用于需对大量字符串进行处理的文字编辑和自动印刷排版系统中。

(8) 向量运算指令

一般在大型计算机或巨型计算机中设置这类指令,可对向量或矩阵进行求和、求积等运算。

(9) 堆栈指令

堆栈是由若干个连续的存储单元组成的先进后出存储区,第一个送入堆栈的数据存放在栈底,最后送入堆栈的数据存放在栈顶。栈底是固定不变的,而栈顶却随着数据的进栈和出栈不断变化,有一个专用的寄存器或指定的存储器单元用来存放栈顶地址,这个寄存器或存储器单元称为堆栈指针 SP。由于堆栈具有先进后出的性质,因而在中断、子程序调用过程中广泛用于保存返回地址、状态标志和现场信息等。

在具有堆栈结构的计算机中,堆栈是提供操作数和保存运算结果的主要存储区,包括算术逻辑运算指令的大多数指令通过访问堆栈获得所需的操作数和保存操作结果。在一般计算机中有压入(堆栈)和弹出(堆栈)两种指令。压入指令(PUSH)把指定的操作数送入堆栈的栈顶,弹出指令(POP)把栈顶的数据送到指令所指定的目的地,并相应地修改堆栈指针。

(10) 输入输出指令

计算机所处理的一切原始数据和所执行的程序(除了固化在只读存储器 ROM 中的以外),均来自输入设备,处理结果需通过输出设备输出,这些通常是由输入输出指令完成的。某些计算机采用输入输出设备和存储器统一编址的方法把输入输出设备中的寄存器看成是存储器的某些单元,任何访问存储器的指令均可访问输入输出设备,因此不再设置输

入输出指令。

(11) 特权指令

某些指令使用不当会破坏系统或其他用户信息,为了安全起见,这些指令只能用在操作系统或其他系统软件中,而不提供给用户使用,称为特权指令。

一般来说,在单用户、单任务计算机中不一定需要特权指令,而在多用户、多任务计算机系统中,特权指令是必不可少的。它主要用于系统资源的分配和管理,包括改变系统的工作方式,检测用户的访问权限,修改虚拟存储器的段表、页表和完成任务的创建和切换等。

在某些多用户计算机系统中,为了统一管理所有的输入输出设备,把输入输出指令也作为特权指令,不允许用户直接使用,当程序需要输入输出时,通过陷阶指令进入操作系统,由操作系统控制输入输出操作。

(12) 控制指令

包括等待指令、停机指令、空操作指令、开中断指令、关中断指令、置条件码指令、置程序状态字指令等。

当用户程序执行完毕后,可安排一条停机指令。但在多用户情况下,不允许停机,因为其他用户程序可能正在等待使用机器,此时可安排一条等待指令或执行只有少量指令的小循环程序(动态停机)。

空操作指令除了将程序计数器增量外,不进行其他操作。

开中断指令、关中断指令一般是进入中断处理程序后用的。

(13) 多处理机指令

在多处理机或多处理器系统中,为了管理共享的公共资源和相互通信,一般设置“测试与设定”或“数据交换”指令,这些指令最主要的特点是在指令执行过程中不允许打断,用以防止多个处理器同时读取和修改共享数据区。

以上涉及的是在计算中通常遇到的指令类型,在每一类型中还可能包括有数量不等、繁简各异的指令,同一条指令还可能有多种寻址方式。随着集成电路工艺的改进和集成度的提高,计算机应用范围的不断扩大,计算机的体系结构不断发展,指令系统也会发生相应的变化。

(王爱英)

所有指令的集合。又称指令集。程序员用各种语言编写的程序要翻译(编译或解释)成以指令形式表示的机器语言以后,才能在计算机上运行。指令由规定计算机操作类型及操作数地址的一组字符组成,在计算机内部用二进制码表示。计算机硬件完成各条指令所规定的操作,并保证按程序所规定的顺序执行指令,所以指令系统反映了计算机的基本功能,是硬件设计人员和程序员都能见到的机器的主要属性。

早期,由于组成计算机的元器件价格较贵,为了降低成本,硬件尽量简单,那时计算机的指令系统是很简单的,一些比较复杂的运算或操作由于程序实现,因此计算机的处理速度很慢。

随着元器件的性能提高和价格下降,硬件成本在一个计算机系统中所占的比例不断下降,而软件成本不断上升。为了便于编程和降低软件成本,计算机的指令系统不断扩充,指令功能逐步增强。指令系统的改进围绕着缩小指令与高级语言语句的语义差异和有利于提高操作系统的执行效率而进行。例如高级语言中的实数计算可通过浮点运算指令实现;某些应用需对数组进行运算时可通过向量指令实现;为了便于程序嵌套,设置了调用指令和返回指令;为了便于操作系统的实现和管理,设置了控制系统状态的特权指令、管理多道程序和多处理机系统的专用指令等。

美国 IBM 公司在 1964 年推出 IBM 360 计算机时宣布了系列计算机的思想。从此以后,各计算机公司生产各自的系列计算机。系列计算机的特征是指令系统、数据格式、I/O 接口等保持相同,因而软件兼容。即使新型计算机或高档计算机扩充了指令系统,但仍保持软件向上兼容的特点,因此保护了用户在软件上的投资。

但是日趋庞大的指令系统不但使计算机的研制周期变得很长,实现起来也很复杂,有时还会降低系统性能。1975 年 IBM 公司提出精简指令集的想法,后来加州伯克莱大学的 RISC I 机和 RISC II 机、斯坦福大学的 MIPS 机研究成功,对精简指令集计算机的发展起了很大作用。此后将过去的指令系统比较庞大的计算机称之为复杂指令集计算机。

参考文献

1. 王爱英主编. 计算机组成与结构(第三版). 北京: 清华大学出版社, 2001
2. 孙强南, 孙显东. 计算机系统结构. 北京: 科学出版社, 1992

zhiling xitong

指令系统(instruction set) 一台计算机中的

3. 李学干, 苏东庄. 计算机系统结构. 西安: 西安电子科技大学出版社, 1991 (王爱英)

zhiwen shibie

指纹识别 (fingerprint recognition) 计算机利用人手指上的条状纹路信息自动识别其身份的过程和技术。指纹是人手指表面的皮肤凹凸不平产生的纹路, 手指皮肤的凸起部分称为脊, 凹陷的部分称为谷。脊和谷的不同分布, 形成了不同的指纹结构特征。

指纹作为生物特征的身份鉴别方式, 具有稳定性和惟一性两大优点。首先, 指纹具有很强的相对稳定性。指纹在人体胚胎发育的第 3~4 月开始生长, 第 6 个月完全形成。指纹一旦形成, 尽管随着年龄变化, 在外形大小、纹线粗细上会产生一定的变化, 而且局部纹线也可能出现新的特征, 然而其纹线的类型、结构、统计特征的总体分布等, 却是稳定的, 没有明显的变化。此外, 对于外界伤害的影响指纹也是相对稳定的。比如手指皮肤受伤, 只要不伤及真皮层, 伤愈以后的指纹纹线仍能够恢复原状; 如果伤及真皮层, 伤愈以后的伤疤则形成新的稳定特征。

其次, 指纹具有惟一性。指纹的形成依赖于胚胎发育时的环境。每个人的指纹都是独一无二的, 而且即使同一个人的十个手指的指纹也有明显的区别。从指纹的特征来看, 每个指纹一般有 70~150 个基本特征点。从概率的角度, 如果两枚指纹中有 12~13 个特征点吻合, 则认为是同一指纹。而按照现有的人口计算, 上述概率需要 120 年才可能出现两枚完全相同的指纹。

指纹识别技术利用指纹纹路的结构特征来进行分类识别。一般指纹纹路的结构特征可以分为全局结构的纹形特征和局部结构的细节特征。指纹识别时首先利用全局纹形特征进行粗分类, 然后再利用局部细节特征进行匹配。

从指纹的全局结构来看, 有如下一些纹形结构: ①弓形, 包括平弓形、帐弓形; ②箕形, 包括放射性箕形、尺骨状箕形; ③斗形, 包括平斗形、中心对称箕、双箕形等。图 1 举例说明了几种典型的纹形结构。

指纹的局部结构特征主要是指指纹纹路的端点、分叉点等, 称为细节特征, 如图 2 所示的小桥、环、分叉点、三角点、端点等。

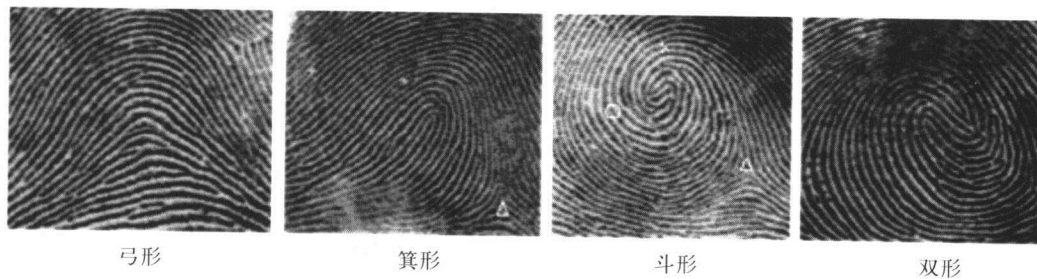


图 1 几种典型的纹形结构

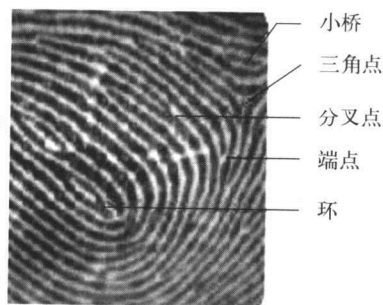


图 2 指纹的细节特征

的细节特征。其基本思想是首先提取指纹纹路上的分叉点和端点, 并记录其位置和方向, 然后按照最优准则, 确定待识别指纹的特征数据相对于指纹模板的平移和旋转, 然后将其对准, 计算匹配上的特征点的个数、比例等参数, 据此来判断待识别指纹是否与指纹模板匹配, 从而确定识别结果, 如图 3 所示。

指纹识别作为个人身份识别最常用的技术之一, 其算法已经比较成熟, 并取得了较好的识别效果。随着人们对于安全性和鲁棒性的要求不断提高, 现阶段的指纹识别在保证安全的前提下普遍存在错误拒绝率较高的问题。如何解决这个问题, 是今后需要进一步关注和研究的内容。

在识别时通常主要使用分叉点和端点两个主要

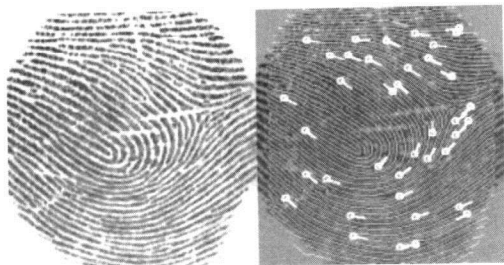


图3 指纹细节特征的提取
(左为指纹原图,右为提取的纹线和特征点图)

参考文献

1. <http://www.sinobiometrics.com/>
2. Jain A and Pankanti S. Automated Fingerprint Identification and Imaging systems. In: Advances in Fingerprint Technology. Elsevier, New York, second edition, 2001
3. Jain A, Hong L, and Pankanti S. Biometrics: Promising Frontiers for Emerging Identification Market. Comm. ACM, 91 ~ 98, February, 2000

(吴志勇 蔡莲红)

zhizhen leixing

指针类型 (pointer type) 由单一的空值及一组标识值组成的数据类型。又称指引元类型。每个标识值标识一个对象,被标识的对象的类型一般是确定的,也可以不确定(如 PL/I 语言)。被标识的对象和标识它们的值是在程序运行时创建和取消的。值 nil 是一个指针常量,它表示不指向任何对象(或说指向空)。它和任何指针类型都是赋值相容的。

PASCAL 语言中,指针类型定义为

类型名 = ↑ 对象类型

其中,对象类型可以是任何类型,在实际应用中一般为记录类型。两个指针类型一致,指的是它们的对象类型一致。PASCAL 语言允许在定义指针类型时,出现先引用后定义的情形。例如,下列类型定义

```
type pointcomp = ↑ complex;
complex = record
    re, in: real
end;
```

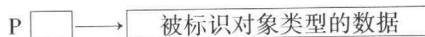
是正确的。

C 语言中指针类型定义为

```
type * name;
```

这里, type(类型)是 C 语言的任何一种有效类型(也叫基类型), name(名字)是指针变量的名称。指针的基类型定义了该指针所指向变量的类型。C 语言中有两个特殊的一元指针运算符: & 和 *。& 的功能是回送其操作数的内存地址。* 与 & 相反,它回送的是存放在其后的地址中变量的值。

指针对象和被标识对象的关系可由下图表示:



在程序说明部分,只说明了指针对象 P,被 P 标识的对象是在程序运行中产生的,所产生的对象的值称为动态对象。动态对象是通过执行语言的预定义过程 new(如 PASCAL, Ada)创建的。其撤销或通过执行预定义过程 dispose(如 PASCAL)或当程序执行到离开其指针对象说明所在的作用域时自动撤销(如 Ada)。

指针类型的操作有相等及不等比较。在程序语言中,一般的动态数据(如栈、链表、队列、树等)是通过指针实现的。

(陈涵生)

zhineng jiqiren

智能机器人 (intelligent robot) 具有人类所特有的某种智能行为的机器。由于“智能”和“机器人”本身尚无明确的定义,因此也可以把智能机器人理解为一类具有高度自主性的自动化机器或设备。智能机器人学是力学、机械学、电子学、控制论、计算机、人工智能和系统工程等多种学科领域相互交叉的边缘学科。

智能机器人是机器人技术发展的高级形态。

在早先的许多科学幻想作品中,机器人是一种能听命于人、能从事劳动、形状像人甚至表现出人的某些思维和行为的人造机器。人类在对这种先进生产工具的最初想象时,就要求它具有人类自身所特有的智能。

遥控操纵器和数控机床的出现为实际机器人的产生提供了技术借鉴。用于危险或有害环境的遥控操纵器主要由两台通过传动机构相联结的多关节机械手臂组成,人操作其中的主机机械手,位于环境现场的从机械手就能跟随复现人手的动作,代替人完成作业。数控机床根据预先储存的工件切削模型以及刀具切削动作序列等数据,对机床的伺服轴进行运动控制,要改变控制只需改变储存的数据。

1954 年,美国的 George. C. Devol 设计并制作了

世界上第一台机器人实验装置,发表了《适用于重复作业的通用性工业机器人》一文,并获得了专利。这台机器人在某种意义上是把遥控操纵器的多自由度关节型连杆机构与数控机床的伺服轴联结在一起,预定的机械手动作一经编程输入后,机械手就可以离开人的辅助而独立运行。此外,操作人员还可以用手带动机械手依次通过作业任务的各个位置,进行“示教”,这些位置序列记录在存储器内。在任务的执行过程中,机器人的各个关节在伺服驱动下重新遍历出那些位置序列,从而完成作业。因此,这台原型机器人的主要技术功能就是“可编程”以及“示教—再现”。

20 世纪 60 年代,机器人产品正式问世,机器人技术开始形成。美国 Unimation 公司和机床与铸造公司(AMF)生产的两种型号的机器人应用于汽车生产线,进行传送、上下料、焊接、喷漆等作业,表现出它有很好的灵活性并显示出明显的经济效益,推动了机器人的商品化。同时,在研究领域也不断地把机器人技术引向深入,如美国麻省理工学院 Lincoln 实验室研制的一种机器人装置配有接触传感器,并与计算机相连,机器人可以凭触觉判断操作对象的形状;美国 Stanford 人工智能实验室研究了如何实现具有人的手、眼、耳功能的计算机系统。美国 Stanford 研究所利用带有电视摄像机和接触觉装置的移动小车,研究了机器人规划问题。上述这类研究工作逐渐形成了智能机器人的技术内容。

70 年代以来,机器人产业蓬勃兴起,机器人技术发展为专门的学科。1970 年,第一次国际工业机器人会议在美国举行,各种成功的实用范例推动了机器人应用领域的进一步发展。日本、瑞典等国家的政府和产业界积极引进机器人技术,建立生产体系,大力推广应用,对机器人的发展具有较大影响。大规模集成电路技术的迅速发展以及微型计算机的普遍应用使得机器人的控制技术出现了质的飞跃。计算机在可靠性、信息加工能力等方面的高度发展还使得人工智能的技术方法在机器人智能化的研究领域中得到实现的可能。60 年代后期,机器人的智能行为一般可以演示积木的分类、堆放动作等。进入 70 年代,研究重点开始转向工业应用。例如,配有视觉、触觉的机器人可以用于铸件、泵体的识别和检查;用于集成电路的装配和焊接。智能机器人的研究范围还扩展到对环境的识别,对景物的理解,任务规划以及人机之间的自然语言对话等课题。模拟人或动物的行走、动作的仿生机械等研究非常活跃,

从仿生学的角度研究生物视觉、触觉、听觉的实现方法也有不少实验成果。

一般认为,按照机器人从低级到高级的发展程度,可以把机器人分为三代。

第一代机器人,即工业机器人,主要指只能以“示教—再现”方式工作的机器人。这类机器人的本体是一只类似于人的上肢功能的机械手臂,末端是手爪等操作机构。由于它仅仅通过操作人员“手把手”的示教,或者通过离线编程预先存储的动作顺序信息来自动完成重复性的作业,因而通常认为不具备智能。

第二代机器人是指基于传感器信息来工作的机器人。它依靠简单的感觉装置获取作业环境和对象的简单信息,通过对这些信息的分析、处理,作出一定的判断,对动作进行反馈控制。这类机器人也称智能型机器人。

第三代机器人,即智能机器人,这是一类具有高度适应性的有一定自主能力的机器人。可以这样来完整地描述智能机器人:它本身能感知工作环境、操作对象及其状态;能接受、理解人给予的指令;并结合自身认识外界的结果来独立地决定工作规划,利用操作机构和移动机构实现任务目标;还能适应环境的变化,调整自身行为。

由上可见,智能机器人是工业机器人从无智能发展到有智能,从低智能水平发展到高度智能化的产物。

区别于第一代、第二代机器人,智能机器人必须具备四种机能:行动机能——施加于外部环境和对象的,相当于人的手、足的动作机能;感知机能——获取外部环境和对象的状态信息以便进行自我行为监视的机能;思维机能——求解问题的认知、推理、记忆、判断、决策、学习等机能;人机交互机能——理解指示命令、输出内部状态、与人进行信息交换的机能。简言之,智能机器人的“智能”特征就在于它具有与外部世界——环境、对象和人相协调的工作机能。

围绕上述四种机能,智能机器人的主要研究内容如下:

(1) 操作与移动 随用途而异的各种机械臂的结构及其驱动方式,包括具有冗余自由度的机械臂以及主从机械臂;多指手爪或其他灵活的操作机构;轮式、履带式、单足跳跃或多足步行式、喷气或气垫式以及仿生式(如蠕动、爬壁)等各种适合不同环境的移动机构及其驱动方式;导航装置和监控技术。

(2) 传感器及其信息处理 基于摄像机的视觉传感器以及对二维、三维物体的信息处理;基于超声波、红外光、激光、雷达波、磁性装置等各种非接触传感器及其定位、识别、运动检测等信息的处理;各种原理的触觉、压觉、力觉、滑觉传感器及其成像、感受、检测等信息的处理;对上述各种传感器信息的解释和理解方法;分析、综合多种传感器信息以便作出合适估计和决策的传感器融合技术。

(3) 控制 对应于各种末端操作机构形式的运动学、动力学建模,运动轨迹规划与生成以及基于运动学或动力学的控制方法;位置控制、力的控制以及力与位置二者结合的柔顺控制方法;遥控作业等非结构环境下的各种主从控制方式;多机器人的协同控制问题;专家控制、模糊控制、人工神经网络控制等新颖的智能控制方法;上述各种控制方法的仿真系统。

(4) 人机交互 用于离线示教和机器人自主行动的各种计算机控制与编程语言,包括动作水平级、对象物水平级和作业目标水平级(任务级)等三类;语音识别与生成、音响识别与检测;声、图、文的多媒体处理技术;遥现视觉、力觉、触觉、声觉的临场感技术及其他虚拟现实技术。

(5) 体系结构 基于机器人整体工作原理,设计它的内部结构组成及其相应的管理控制方式。例如分层递阶式体系结构——最上层的组织级将给定的外部命令和任务分解为子任务或动作的组合,传至下一层的协调级,产生一系列具体动作序列,再传至最下层的执行级形成行动机构的驱动指令,而各种外部传感器信号逐级向上反馈,经综合处理后影响决策。又如基于行为的包容结构——若干基于传感的可以自由组合的独立单元直接建立从感知到控制动作间的映射关系,系统的行为由单元行为的时间、空间序列组成,并由仲裁机构监控。

(6) 机器智能 上述研究内容都涉及到机器智能问题。具体的技术和方法主要有问题求解、自动规划生成、模式识别、自然语言处理、机器学习、专家系统、知识库等,另外还涉及到人类智能的机理实质和人工智能的实现途径这两方面的基础理论研究。

(7) 应用研究 智能机器人的材料、能源等实用化研制问题;各行各业对智能机器人的应用需求分析和产业发展关系;智能机器人的实用所引起的人类社会心理和法律问题等。

智能机器人与人工智能学科密切相关,它一方

面是人工智能技术与方法的典型应用对象,另一方面又是研究、开发人工智能技术与方法的实验床,其核心问题是如何研究、实现“从感知到行动的智能联系”。计算机科学与技术的发展为智能机器人的工程实现提供了支持和推动力,智能机器人也可以看成是安装了各种拟人的或仿生的外部设备的计算机系统或装置。智能机器人与工业机器人或智能型机器人由于满足不同的应用需要而可同时并存,但智能机器人实际上研究的是各种形态的智能机器,具有更为深入的学术意义和更为广泛的应用前景。

目前,智能机器人的研究还处于初级阶段,研究目标一般围绕感知、行动、思考三个问题。实验室原型主要有:自动装配机器人——具有对部件的三维视觉识别和定位,柔顺控制多指手爪的抓取和精密装配,自动规划装配序列,避碰,多操作器协调等功能;移动式机器人——具有室内、外自主导航,路径规划,避碰,野外、壁面环境下的移动,基于感觉的取样操作和检测、排除故障等功能;水下机器人——具有深水潜游,有缆遥控,水下清理、维修或敷设等功能。

上述各种机器人实际上只是理想的、完整的智能机器人的部分功能环节,研究的局限性主要在于人工智能技术至今还不能提供实现机器智能的有效原理和方法。智能机器人的完美实现需要提高机器的自主性——进一步独立于人,建立更友善的人机界面;需要提高机器的适应性——更好地适应环境变化,加强与环境的交互能力。

智能机器人的研究目前正在三个方面深入:依靠人工智能基于领域知识的成熟技术,发展面向专门任务的特种机器人;在研制各种新型传感器的同时,发展基于多传感器集成的大量信息获取和实时处理技术;改变排除人的参与、机器人完全自主的观念,发展人机一体化的智能系统。

参考文献

1. 周远清,张再兴,许万雍,贾培发. 智能机器人系统. 北京:清华大学出版社,1988
2. 张钊. 智能机器人的现状及发展. 科学导报, 1992,6
3. 刘海波,李根深. 智能机器人研究现状及发展动向. 机器人情报,1991,5

(张再兴)

zhinengka yueduqi

智能卡阅读器(smart card reader) 从智能

卡上读出或写入数据的设备。

智能卡也叫 IC 卡,即集成电路卡,是在塑料基片上嵌入专用半导体集成电路芯片的卡。常见的 IC 卡大小为 85.6 mm × 53.98 mm,厚度为 0.76 mm。移动通信系统使用尺寸较小的用户身份组件(SIM)卡,其大小为 25 mm × 15 mm。

IC 卡由半导体芯片、塑料基片及通信接口组成。依照卡与外界数据交换的形式,IC 卡可分为接触式、非接触式和双界面三种类型。接触式 IC 卡通过卡上的电极触点与外界直接接触来交换数据,这种卡的通信接口为电极膜片。非接触式 IC 卡通过电磁波与外界进行数据交换,这种卡的通信接口为天线及相关电路。双界面 IC 卡同时拥有以上两种通信接口。

根据卡中半导体芯片功能的不同,IC 卡主要分为存储卡、逻辑加密卡和 CPU 卡。存储卡只有数据存储功能。逻辑加密卡是在存储卡的基础上增加安全控制逻辑,这在一定程度上可保护卡中数据的安全。CPU 卡内集成了中央处理器、存储器、加密协处理器、输入输出接口电路等,这种卡中一般固化有卡内操作系统(COS)。

作为一种有效的身份识别工具和支付手段,与磁卡相比,IC 卡具备以下主要特点:具有多层次的安全认证功能,不易伪造或复制,存储容量大,寿命长,抗电磁干扰能力强。

IC 卡阅读器是一个单片机系统,主要由 IC 卡接口电路、中央处理器(CPU)、CPU 外围扩展模块、加解密模块、电源模块及通信接口组成。IC 卡接口电路完成对 IC 卡的读写驱动。CPU 外围扩展模块可以是存储器、时钟电路、键盘电路及显示电路等。加解密模块可由专用加密协处理器构成。IC 卡阅读器与接触式 IC 卡及非接触式 IC 卡交换数据的接口分别是 IC 卡座和天线,与其他设备如计算机进行通信一般通过 RS-232C 或 USB 接口完成。

IC 卡阅读器从应用领域上可分为专用型与通用型。专用型 IC 卡阅读器通常是作为相关设备的一部分,并且在结构上是按一体化设计的,如 IC 卡电话机、IC 卡电表、IC 卡门锁等。通用型 IC 卡阅读器可独立使用(独立型)或作为终端使用(联机型)。独立型 IC 卡阅读器拥有键盘和显示屏等人机界面。这种阅读器可自带微型打印机及网络接口。由于操作形式不同,它可分为台式和手持(便携)

式。台式 IC 卡阅读器注重牢固、耐用,手持式 IC 卡阅读器强调轻巧、省电。联机型 IC 卡阅读器分为内置式和外置式。内置式 IC 卡阅读器通常安装在主机前面板的安装槽上,它可通过扩展总线与计算机进行通信。外置式 IC 卡阅读器一般放在工作台上使用,易于安装。

作为一种专用输入设备,IC 卡阅读器具有操作安全可靠、使用方便的特点。随着 IC 卡在电信、金融、税务、保险、交通、医疗、卫生保健、城市公用事业等多个领域的普及,IC 卡阅读器可得到更广泛的应用。

参考文献

王卓人,邓晋钧,刘宗祥. IC 卡的技术与应用. 北京:电子工业出版社,1999 (李炜)

zhibiao yuyan

置标语言(markup language) 用于描述一组“置标”标记符号,并相应地给出它们放置到电子文档内的表示形式和放置位置的语言规范。“置标”标记是指有别于电子文档内容正文的任何形式的记号和标志。它们可以用来指示文档中的章节分段、印刷时的排版布局格式、或用于指出一段正文内容的引文出处等。一般地,“置标”标记是指为解释信息内容而放置在文档正文中的任何标记。置标语言应该能够指明标记区别于正文的手段,给出标记的格式以及它们的含义和用途等。

20 世纪 70 年代,首次提出了通用置标语言(GML),它是置标语言的雏形。引入了严格定义具有嵌套元素结构文档的机制,制定了在文档内嵌入描述性标记的标准格式。通过这些机制,GML 可以用来定义具体的置标语言。因此,GML 不是一个具体的置标语言,而是一个定义具体的置标语言的语言。从 1978 年开始,国际标准化组织 GCA、ANSI 和 ISO 等,在 GML 的基础上引入了一些新概念。在 1986 年形成了通用置标语言标准(ISO 8879)语言,称作**标准通用置标语言(SGML)**。

SGML 语言为电子出版和其他数字信息产业提供了便利,制定标准化的置标格式和相应的语义,可以使文档的置标独立于处理文档的各类机器平台和处理程序。SGML 在出版等信息产业中已经得到广泛应用。

万维网所使用的一种置标语言称作**超文本置标**

语言(HTML)。由于置标语言在各行各业的信息加工的重要性,并为了满足不同行业信息加工的不同需要,近年来,万维网联盟(W3C)国际组织的一个工作组在Jon Bosak的领导下制定了可扩展置标语言(XML)。

参考文献

1. Charles F Goldfarb, Edward J Moshere, Theodore I Peterson. An Online System for Integrated Text Processing. Proc. American Association for Information Science 7, 1970, 147 ~ 150
2. SGML/XML Web Page. <http://xml.coveragers.org/>, or <http://www.oasis-open.org/cover/>
3. W3C HTML Home Page. <http://www.w3.org/MarkUp/> (瞿裕忠)

zhongduan

中断 (interrupt) 计算机在执行程序过程中,当遇到急需处理的事件时,暂停当前正在运行的程序,转去执行有关服务程序,处理完后自动返回原程序,这个过程称为中断。

中断可分为内中断和外中断。内中断是由计算机内部原因引起的中断,如溢出中断、非法操作码中断、地址越界中断等;外中断指外部事件引起的中断,如输入输出中断、电源故障中断、实时时钟中断等。外中断又叫强迫中断。在内中断中,由程序中特设的指令引起的中断,又称为软中断。

要求中断的请求可按其轻重缓急分级,并赋予一定的优先权,称为中断优先级。当有多个中断请求时,中断系统按中断优先级进行排队。排队原则是:级别高的优先响应。若在处理低级中断过程中又有高级中断申请中断,则高级中断可以打断低级中断处理,转去处理高级中断,等处理完高级中断后再返回处理原来的低级中断,称为中断嵌套。为了增加中断排队的灵活性,还可用程序的方法在某段时间中屏蔽某些中断请求,以改变中断响应顺序。有些中断请求是不能屏蔽的,如电源一旦掉电,中央处理器应立即响应,其优先级最高,称为非屏蔽中断。

中央处理器在响应中断后转入具体的中断服务程序之前必须保存其现场,包括程序断点、程序状态字和运算器中通用寄存器内容,以保证中断服务后能够恢复现场返回原来的程序。在保存现场和恢复

现场的阶段,不允许任何新的或更高级的中断打断。系统采用“关中断”的办法禁止响应任何中断,等到保存现场或恢复现场完毕,再“开中断”。中断处理过程包括保存现场、恢复现场和具体的服务处理,都是通过程序实现的,因此这种方式又叫程序中断方式。

为了提高响应中断的速度,通常把所有中断服务程序的入口地址(或称中断向量)汇集为中断向量表。当中央处理器响应中断时,从中断向量表中直接得到相应的入口地址,并从该地址开始执行中断服务程序。

中断在现代计算机系统中是一种非常重要的技术,输入输出设备和主机交换数据、分时操作、实时系统、多处理机系统、计算机网络和分布式计算机系统中都要用到这种技术。

参考文献

- 谢树煜,刘风云. 计算机概论. 北京:机械工业出版社,1987 (谢树煜)

zhongguo youlu wenti

中国邮路问题 (Chinese postman problem)

图论中一个有重要理论意义和广泛应用背景的问题。它来源于下述实际问题:“一个邮递员应如何选择一条路线,使他能从邮局出发,走遍他负责送信的所有街道,最后回到邮局,并且所走的路程为最短。”归结为数学问题,则是:“设给出了一个连通的无向图,它的每条边都有非负的长度,求 G 的一条经过每条边至少一次并且总长度最小的闭路径。”

上述问题是中国学者管梅谷于1960年最早提出并加以研究的,当时称为“最短投递路线问题”,并给出了解这个问题的第一个算法。

1965年以后,国内外学者对中国邮路问题做了许多研究,并对原问题进行了许多推广与变形,以至于现在提到中国邮路问题时,指的是一大类问题,包括:

无向图上的中国邮路问题就是指上面提到的中国邮路问题。

有向图上的中国邮路问题指在一个有向图上,求一条经过每条有向边至少一次并且总长度最小的闭路径。这个问题相当于在所有街道都是单行线的情况下,求解最短投递路线问题。

混合图上的中国邮路问题指在一个既有无向边

又有有向边的图上求一条经过每条无向边和有向边至少一次并且总长度最小的闭路径。这个问题相当于在部分街道是单行线的情况下,求解最短投递路线问题。

带风向的邮路问题指在一个无向图上,在下述假设下:“对于以 i 与 j 为顶点的边,从 i 到 j 和从 j 到 i 的长度可以不同”,求一条经过每条边至少一次并且总长度最小的闭路径。

乡村邮路问题指在一个无向图上,给定了一个边的子集,求一条经过给定子集中的每条边至少一次并且总长度最小的闭路径。

还有一些其他的推广,如有向乡村邮路问题,有容量限制的中国邮路问题等。对所有这些问题,均找到了不少算法,并对问题的复杂性进行了分析。已经证明,无向图与有向图上的中国邮路问题有多项式算法,而其他问题都是 NP 困难的。

中国邮路问题在近二三十年中得到了广泛的应用,除用于邮政部门外,还曾用于扫雪车路线、洒水车路线、警车巡逻路线等的最优设计上。另外,从 80 年代起,人们又发现用计算机绘图时,如何节约画笔的空走问题,所对应的数学模型,恰好是中国邮路问题。而在计算机制造工业中,如将激光刻制用于集成电路加工的模具时,也可以用中国邮路问题来减少激光刻印机的工作时间。从而使中国邮路问题在计算机工业中也获得了重要的应用。

参考文献

Minieka E. Optimization Algorithms for Network and Graphs. New York and Basel: Marcel Dekker, Inc., 1978
(管梅谷)

zhongjiqi

中继器 (repeater) 位于开放系统互连基准(参考)模型(OSI/RM)的第一层——物理层的一种网络互连设备(见图1)。它具有对信号进行放大、补偿、整形和转发的功能,主要用来扩展网络电缆的长度,连接分段网络。

电子信号通过传输介质(如同轴电缆、双绞线电缆等)时会发生信号衰减,传播距离越长,衰减越厉害,从而可能引起信号无法分辨而丢失。例如,ISO 8802-3 的 10 Base 5 规范规定以太网的粗缆总长度(网络跨度)可以达到 2.5 km,但由于信号衰减的原因,每个网络段可允许的粗缆最大长度不得大

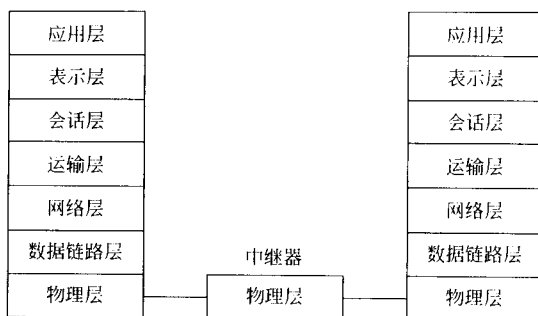


图1 中继器在物理层上使网络互连

于 500 m,这样就限制了整个网络的覆盖范围。

中继器把从一段电缆接收到的信号经过放大、补偿和整形后,转发到另一段电缆上,从而延长了电缆的总长度,扩展了网络的覆盖范围(见图2)。

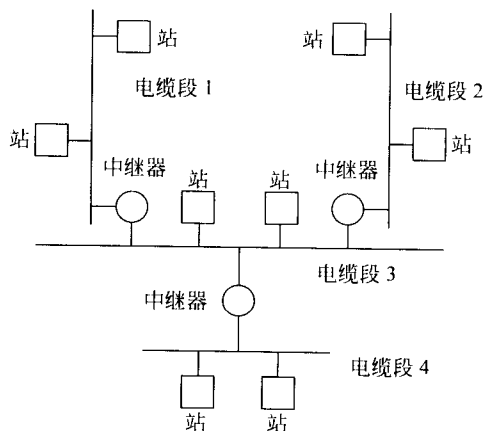


图2 用中继器扩展网络

中继器也可以在两种不同的介质之间转发信号,比如,把信号从基带同轴电缆转发到光缆,从光缆转发到基带同轴电缆等,在这种情况下也被称作收发器。中继器一般为双口,但也有多口的中继器。中继器和中继器之间可以用电缆连接,也可以用光缆连接,后者称为光缆中继器。

中继器是物理层的网络设备。它不能做任何数据过滤工作,而只是简单的把信号整形放大,达到扩展网络的目的。在以太网中如果使用中继器来扩展网络,要注意在拓扑结构上不能形成环路。在建筑物与建筑物之间也可以考虑用光缆中继器来互连网络,这样做既有利于支持较长的传输距离,又有利于

网络安全性,提高网络的抗干扰能力。图3所示用光缆中继器互连大楼间的网络,显然在这种情况下必须成对地使用光缆中继器。

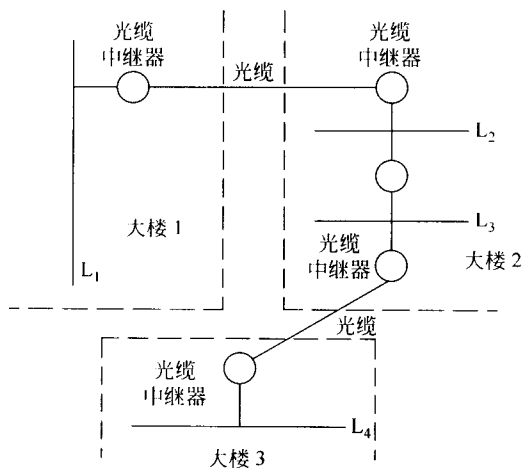


图3 用光缆中继器连接大楼间的网络

除了前面所提到的限制以外,使用中继器扩展网络时,还要注意计算所扩展网络的最大传输延迟,因为中继器本身是有延迟的。另外,还应分析各个网段的负载情况,因为每连接一个网段,就会使全网的整体负载相应增加。(张世永)

zhong ri han tongyi hanzhi

中日韩统一汉字 (Unified CJK Ideographs)

在 ISO/IEC 10646 和 Unicode 中按照一定的认同和甄别规则整合在一起的汉字集合。也称“CJK 汉字”。GBK 和 GB 18030 汉字编码字符集中包含了所有 CJK 汉字。

随着经济全球化趋势的加快,文字的国际性越来越重要了。汉字是一种可以用于多国语言的文字,它用于日语(汉字称汉字, Kanji),韩语(汉字称汉字, Hanja),古朝鲜语(吏读汉字, Lidu),古越南语(字喃或喃字, ChuNom)等。

为了方便地实现多文种信息处理,国际标准化组织 ISO/IEC 和 Unicode 集团在制定通用编码字符集时,遵循汉字的 XYZ 模型,对汉字进行了认同和甄别,不同国家和地区使用的汉字,不论其字音和字义有无区别,只要字形相同,该汉字就只有一个代码。

汉字的 XYZ 模型可以用图1加以说明。图中

X 轴表示字义, Y 轴表示字形(抽象字形), Z 轴表示字型(具体造型)。比如,艺术的艺字。如果按字义认同,即向 X 轴投影,则其简体形式 艺、繁体形式 藝、异体形式 藝、日文略字形式 芸都具有相同的数值(X_1);作为芸香、运薹、芸芸众生意的芸,则具有其他的数值(X_2, X_3, \dots)。如果按抽象字形认同,则无论它们在意义上有何联系,也无论它们在细微末节的造型上有何差异 芸-芸(Z 轴),它们在 Y 轴的投影为 4 个不同的值。中日韩汉字的认同与甄别,即是在抽象字形的基础上进行的。

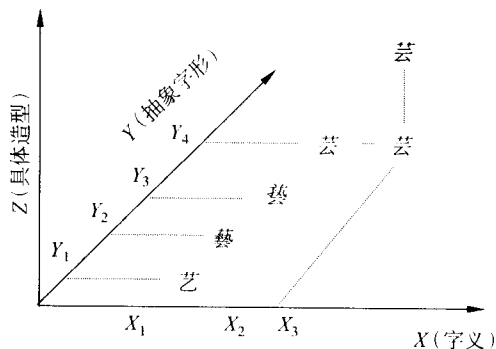


图1 汉字的 XYZ 模型

CJK 统一汉字中的所谓“统一”,并不是强求中、日、韩的汉字书法造型一致,而是将认同的中、日、韩汉字赋予相同的代码,不认同的汉字则赋予不同的代码。即使具有相同代码的汉字,在不同国家和地区仍然可以在认同规则的总框架下保留其书法造型的差异,比如丰 = 丰,草 = 草,道 = 道,骨 = 骨 = 骨,等等。

在 ISO/IEC 10646—1:1993 中一共定义了 20 902 个 CJK 汉字。它们源自中国及中国台湾地区、日本、韩国的 13 个字符集,它涵盖了 GB 2312 之全部,GB 8565 之全部,《现代汉语通用字表》之全部,Big 5 之全部,台湾地区新、旧电报码之全部,日本的 JIS 汉字之全部,韩国 KSC 汉字之全部。这些汉字的字序,是按照汉字在中、日、韩四字典的页码、字位综合排列的。

ISO/IEC 10646—1:2000 版(或 Unicode 3.0)中扩充了 6 582 个中日韩汉字(即 CJK_A),在 ISO/IEC 10646—2:2000 中,又进一步扩充了 42 778 个 CJK

汉字(即 CJK_B)。

参考文献

1. ISO/IEC 10646—1: 2000. Information technology—Universal Multiple-Octet Coded Character Set—Part 1: Architecture and Basic Multilingual Plane
2. ISO/IEC 10646—2: 2001. Information technology—Universal Multiple-Octet Coded Character Set (UCS)—Part 2: Supplementary Planes

(张轴材 张福炎)

zhongwen xinxi chuli

中文信息处理 (Chinese information processing)

利用计算机对汉语的音、形、义及其所带信息进行加工、操作的过程、理论、方法和技术,或以之为研究对象的学科。所谓“加工”、“操作”,包括输入、输出、识别、转换、压缩、存储、检索、分析、生成和理解等。它是在语言文字学、计算机应用技术、人工智能、认知科学、信息论和数学等众多学科基础上衍生出来的一门边缘学科。一方面,汉字学、汉语的词汇学、句法学、语义学、语用学构成了中文信息处理的语言学基础;另一方面,人工智能、信息论和数学中的模型论、形式化理论、模式识别理论、统计学习理论则形成了其计算方法的基础。

我国自 20 世纪 70 年代后期以来,在政府的大力倡导下开展了多项中文信息处理的重点攻关课题,在汉字编码字符集、汉字输入、输出、汉字字型、汉字激光照排、手写体汉字识别、汉语言语(语音)识别和文语转换、信息检索和机器翻译等一批应用系统上取得了重大进展。其中汉字输入和激光照排等技术的成功开发,对我国社会信息化进程和信息产业的崛起产生了巨大影响。在 2000 年—2001 年中国工程院和中国科学院联合评选的 20 世纪中国重大技术成就中,“汉字信息处理与印刷革命”被评为排名在第二位的重大成就。

中文信息处理的研究方法大体上可分为两种。一种把基点放在语言理解上,依靠词法-句法分析和语义解释来实现语言信息处理任务。由于它所依据的语言知识主要来自语言学家的直觉和经验,这种方法又称为理性主义方法。另一种方法是概率统计方法。由于语言信息处理所需的知识不仅规模浩大而且粒度极细,统计论者认为知识获取任务如果没有计算机的积极参与是不可想象的。他们主张,语言知识的直接来源是语料库,是真实文本中可观察到的语言

事实。所以这种方法又叫作经验主义方法。

研究表明,在汉语的字、词、句、篇章等每个层面和音、形、义之间普遍存在着—对多的歧义。这也是各种自然语言的普遍规律。在中文信息处理的不同任务中,一个中心任务便是歧义消解。究竟是用理性主义方法,经验主义方法,还是这两种方法的结合,归根结底取决于它们各自在歧义消解上的效能。

中文信息处理的应用系统,包括汉字键盘输入、听写机(言语识别)、文语转换(言语生成)、手写体汉字识别、跨语言信息检索(搜索引擎)、信息提取、自动文摘,以及各种自动翻译和辅助翻译软件,是当前这一领域中具有重要社会和经济效益的研究开发方向。

参考文献

1. 赵伯璋,徐力. 计算机中文信息处理(上下册). 北京: 宇航出版社, 1987
2. 刘开瑛,郭炳炎. 自然语言处理. 北京: 科学出版社, 1991
3. 黄昌宁,李涓子. 语料库语言学. 北京: 商务印书馆, 2002

(黄昌宁)

zhongwen xinxi jiansuo

中文信息检索 (Chinese information retrieval)

将作为主要信息来源的中文文献资料按一定的方式进行组织、储存、管理,并根据用户的要求查找到所需信息的方法、技术和过程。20 世纪 90 年代前,在中国,“信息检索”称为“情报检索”。由于信息来源不断扩充(从文献资料库到因特网),信息形式和信息载体不断丰富(从书目数据库到超文本、多媒体等),作为同义词的“情报”已让位给“信息”。国际上,信息检索的探索和试验始于 20 世纪 50 年代,60 年代已经实用化,70 年代联机检索服务形成市场,80 年代努力实现多元化、智能化,90 年代信息检索系统与因特网结合,搜索引擎技术迅速发展。我国的中文信息检索研究始于 70 年代著名的“七四八”工程。80 年代初开始提供国际联机检索服务。此后,我国自主开发的数据库(文献库当然是中文占压倒多数)和联机检索服务系统迅速增加。全文检索技术的发展缩小了中文信息检索和西文信息检索的差距。

如果实现技术以字符串匹配为基础,中文信息检索和西文信息检索在原理和机制上并无本质区别。当今多文种处理已成为计算机系统普遍具有的能力,中文信息检索和西文信息检索实际上已经

融合。但是,当信息检索技术与自然语言处理技术相结合向更高层次的智能化的概念检索发展时,汉语信息处理的一些特点和难点同时也开拓了中文信息检索技术发展的空间。例如,正确地实现中文文本的词语切分和词性标注不仅可以压缩索引存储量、提高检索速度,而且可以改善系统的性能指标。

中文信息检索主要研究内容有:①信息检索建模,即采用何种方法表示文档和检索要求并计算它们之间的相关性。主要的模型有布尔模型、向量空间模型、概率模型、扩展布尔模型等,其中布尔检索模型为大多数商用系统采用。②文献处理,主要指自动标引、自动分类和自动文摘。③基础资源建设,包括停用词表和主题词表的构造。④实现技术,包括倒排文件结构、位图文件、散列索引、B 树索引等快速检索技术。⑤检索效果评价体系,其中查全率(检出的相关文献量与系统文献库中的相关文献总量之比)和查准率(检出的相关文献量与检出的文献总量之比)是最重要的两个评价指标。⑥汉语自动分析技术及其与中文信息检索技术的结合。

实用的中文信息检索系统主要是书目型的标引检索系统和可用文章中的字串或自由词进行检索的全文检索系统。中文信息检索系统的用户包括政府部门、报社和新闻社、信息中心、图书馆、法律部门、公司乃至个人。

Internet 的发展将信息检索技术发展到了搜索引擎技术。Internet 搜索引擎除了需要有全文检索系统之外,还要有所谓的“蜘蛛”系统,即能够从互联网上自动收集网页的数据搜集系统。“蜘蛛”将搜集到的网页的内容交给索引和检索系统处理。当然,一个完整的搜索引擎还需要有检索结果的页面生成功能。

参考文献

1. Frakes W B, Yates R B. Information Retrieval: Data Structures and Algorithms. Prentice Hall, 1992
2. 赖茂生等. 计算机情报检索. 北京: 北京大学出版社, 1993
3. 李晓明, 李星. 搜索引擎与 Web 挖掘进展. 北京: 高等教育出版社, 2003 (俞士汶)

zhongyang chuliqu

中央处理器 (central processing unit, CPU)

在计算机内部对数据进行处理并对过程进行控制的部件。中央处理器由运算器、控制器和处理器总线等组成。

指令执行过程

计算机在程序控制下进行信息处理。程序由指令组成。指令也像数据那样编码并存放在存储器中。程序运行时按控制器中的程序计数器所指地址,从存储器读出指令,经过指令译码器等部件的分析,向计算机各部件发出各种操作命令,完成相应指令的功能,最后得到程序运行的结果。

典型的算术逻辑指令的执行过程是:①形成指令地址,把指令地址送给存储器,发读存命令,读出本条指令。②取出指令送到控制器的指令寄存器,对指令操作码进行译码,决定执行何种运算或操作。③根据寻址方式形成操作数地址,把操作数地址送给存储器,发读存命令,读出操作数,送入运算器。④若需两个操作数可重复步骤③。⑤在运算器中对操作数进行操作码指定的运算。⑥形成运算结果地址,把结果地址和运算结果送给存储器,发写存命令,写入结果。若为寄存器-寄存器型指令,操作数在寄存器中,操作结果也放在寄存器中,除了取指令外不再访存,大大缩短了执行指令的时间。

控制器

控制器用来生成指令地址,取出指令,分析指令,向各部件发出一系列有序的操作控制命令,实现指令功能。在机器发生意外故障,或者有随机的输入输出请求时,采用中断方式终止现行程序的执行,转入中断处理程序,处理完随机请求后,又自动返回原来的程序。

同步控制与异步控制 计算机在执行指令的过程中,各种控制信号的时间关系是非常严格的,必须在准确的时间给出控制信号。控制信号如何产生,操作持续多长时间,下一个操作如何启动等问题,属于计算机的控制方式问题。

异步控制 当前的操作完成时,利用其完成信号作为后一操作的启动信号,这种由操作部件自己决定处理时间节奏的时序控制方式称为异步控制。

同步控制 控制器按固定的时间节拍给出控制信号来执行各种操作,这种控制方式称为同步控制。

如果各种操作时间相差不悬殊,而且又不随机变化,可采用同步控制方式。其特点是控制简单,易于实现。但为了照顾个别操作时间特别长(如乘法、除法运算),则可以局部地采用异步控制,但从

全局看仍是同步控制。

控制器的基本组成 控制器包括:指令部件、地址部件、时序部件、中断控制部件和操作控制部件。

指令部件 指令部件包括:指令计数器(或程序计数器)——给出要执行的指令地址,保证程序自动连续地执行;指令寄存器——存放正在执行的指令,供控制器分析解释,直到该条指令执行完毕;指令译码器——把指令操作码翻译成某一个控制线上的有效电位,控制完成有关的操作。

地址部件 根据寻址方式形成转移地址或操作数有效地址。通常包括地址加法器,如在变址寻址时,把指令中的位移量与变址寄存器的内容相加,得到需要访问的主存地址。有时为节省器件或简化线路,也可将地址加法交运算器的算术逻辑部件 ALU 完成。

时序部件 给出有一定顺序时间关系的信号。一条指令执行过程可分为几个阶段,如取指令、执行等。每个阶段又可分成几步,每一步叫作一个节拍。节拍是时序信号的基本单位。节拍周期通常与时钟周期一致。启停电路用来实现启动、停机和单条、单拍操作,并控制这些操作时的时钟信号的输出。

中断控制逻辑 可以处理随机出现的各种意外请求,包括低速输入输出设备的输入输出请求。它不但提高了机器的工作效率,而且提高了机器的可靠性,并为分时操作、实时控制和网络通信提供了实现手段。

操作控制部件 是运行程序和执行指令的关键部件。为了指挥计算机各个部分协同工作,完成指令规定的功能,操作控制部件必须准确及时地向各部件提供各种操作控制信号。操作控制部件可以采用组合逻辑电路来实现,称为**硬连线控制器**。这种控制器的维护、调试、修改、扩充指令等都很困难,但因为速度比较快,为高性能计算机和精简指令集计算机所采用。除了硬连线控制器外,在很多计算机中采用微程序控制器(参见**微程序控制器**)。

运 算 器

运算器是计算机的数据处理核心部件。主要由执行算术运算和逻辑运算的部件、存放操作数和中间结果的寄存器组以及连接各部件的数据通路组成。

串行运算器与并行运算器 早期的计算机是用

分立元器件组成的,其线路复杂,价格昂贵。为了简化线路和降低成本,把运算器设计成串行的,即只设置一位二进制全加器,每一个数的各位二进制数位依次由低位到高位进入加法器中形成运算结果。现在除了简单的单片机和计算器之外,很少采用串行运算器。

并行运算器对参与运算的二进制数的每一位同时进行运算。通常所说 32 位计算机,是指参与运算的二进制数是 32 位。这种计算机的并行运算器应是 32 位全加器。所有的寄存器、存储器和数据通路也都是 32 位的。显然,并行运算器的运算速度比串行运算器的高。并行加法器中最后结果的形成与进位有关。由于进位是由低位到高位逐位向前串行传递的,特别是当计算机字长较长时,加法运算时间将因进位原因而增加。为了提高加法速度,提出了多种快速进位的方法。

串行进位电路与并行进位电路 两个二进制数相加,若不考虑低位来的进位,相应位相加得到的结果称为半加和。但是两个数相加要得到正确的结果,必须考虑低位来的进位,这样得到的结果称为全加和。高位全加和的形成必然依赖于低位进位的到来,这种进位由低位到高位逐位串行形成的加法器叫串行进位加法器,又叫**行波加法器**。例如两个二进制数: $A = A_n A_{n-1} \cdots A_4 A_3 A_2 A_1$, $B = B_n B_{n-1} \cdots B_4 B_3 B_2 B_1$, 利用串行进位加法器求和,其每一位的全加和及进位的逻辑表达式如下:

$$\text{全加和 } S_i = A_i \oplus B_i \oplus C_{i-1}$$

$$\text{进位 } C_i = A_i B_i + (A_i + B_i) C_{i-1}$$

$$\text{最低 4 位全加和} \begin{cases} S_1 = A_1 \oplus B_1 \oplus C_0 \\ S_2 = A_2 \oplus B_2 \oplus C_1 \\ S_3 = A_3 \oplus B_3 \oplus C_2 \\ S_4 = A_4 \oplus B_4 \oplus C_3 \end{cases}$$

其中 C_0 为最低位需要加上 1 时的信号。

$$\text{最低 4 位进位} \begin{cases} C_1 = A_1 B_1 + (A_1 + B_1) C_0 \\ C_2 = A_2 B_2 + (A_2 + B_2) C_1 \\ C_3 = A_3 B_3 + (A_3 + B_3) C_2 \\ C_4 = A_4 B_4 + (A_4 + B_4) C_3 \end{cases}$$

可以看出每一位的进位及每一位的全加和都是由低位向高位逐位串行形成的。因此,字长越长,加法时间越长。

为了提高加法速度,提出了并行进位的方法,使每一位的进位形成不必等待低位进位。其办法是将上述逻辑表达式代入,改写得到:

$$\begin{cases} C_1 = A_1 B_1 + (A_1 + B_1) C_0 \\ C_2 = A_2 B_2 + (A_2 + B_2) A_1 B_1 \\ \quad + (A_2 + B_2) (A_1 + B_1) C_0 \\ C_3 = A_3 B_3 + (A_3 + B_3) A_2 B_2 \\ \quad + (A_3 + B_3) (A_2 + B_2) A_1 B_1 \\ \quad + (A_3 + B_3) (A_2 + B_2) (A_1 + B_1) C_0 \\ C_4 = A_4 B_4 + (A_4 + B_4) A_3 B_3 \\ \quad + (A_4 + B_4) (A_3 + B_3) A_2 B_2 \\ \quad + (A_4 + B_4) (A_3 + B_3) (A_2 + B_2) A_1 B_1 \\ \quad + (A_4 + B_4) (A_3 + B_3) (A_2 + B_2) (A_1 + B_1) C_0 \end{cases}$$

由以上逻辑关系式可以看出, C_1, C_2, C_3, C_4 直接决定于输入的两个相加数 A, B 和 C_0 , 与中间形成的低位进位无关。同理, 也可写出 $C_5, C_6 \dots$ 的逻辑表达式。这些表达式都是与、或项, 因此可用与门、或门电路实现。采用这样的方案的加法器称为并行进位加法器。又称先行进位加法器。

采用这种方案时, C_4 的形成共有 5 个“与”项相“或”, 其中最后一个“与”项有 5 个输入变量。当字长为 n 时, 显然需要 $(n+1)$ 个“与”项相“或”。而输入变量最多的“与”项共有 $(n+1)$ 个输入变量。在“与”门及“或”门的输入端数受到限制的情况下, 这种完全并行的并行进位加法器要实现是很困难的。

一种折衷的办法叫分组并行进位方案。即把整个加法器分成若干小组 (例如 4 位 1 组), 小组内利用上述办法实现并行进位, 小组间仍为串行进位。为了进一步提高加法速度, 组间进位也可采用类似的方法实现并行进位, 这就产生了多级分组并行进位方案。

定点运算器与浮点运算器 定点运算器完成两个定点二进制小数的算术运算和逻辑操作, 操作数的小数点位置是固定的、隐含的, 位于最高数值位之前, 符号位之后。定点运算器的特点是结构简单, 但数据表示范围小, 不适合科学计算。

定点补码加法器 其基本结构框图如图 1 所示。

A 和 B 为两个 $(n+1)$ 位的寄存器, 存放参加运算的操作数。 Σ 为全加器, 共 $(n+2)$ 位, Σ_{n+1} 与 Σ_n 为双符号位, 判断结果溢出用。运算结果存入 A 中。

执行加法操作 $A+B$ 时, 控制器发出命令 $A \rightarrow \Sigma$, $B \rightarrow \Sigma$ 和 $\Sigma \rightarrow A$, 加法即可完成。当作减法 $A-B$ 时, 控制器提供 $A \rightarrow \Sigma$, $\bar{B} \rightarrow \Sigma$ 和 $\Sigma+1$, $\Sigma \rightarrow A$, 减法即可完成。其中 $\Sigma+1$ 是指加法器末位加 1, 因为作减 B 时, 实际上是作加 \bar{B} , 再在末位加 1 实现的。

作乘法时, 运算器中最少还需增加一个乘商

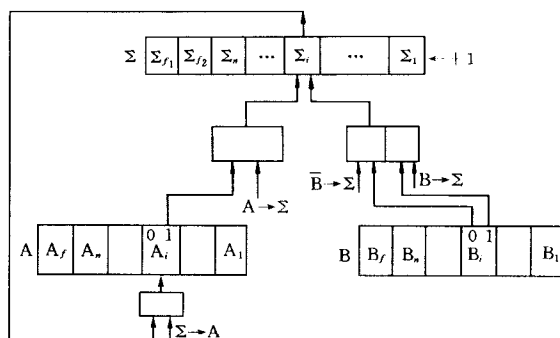


图1 定点补码加法器的基本结构框图

寄存器。乘法时用来存放乘数及低位乘积; 除法时用来存放商。另外, 还需增加移位线路。典型的定点原码乘法器的原理框图如图 2 所示。

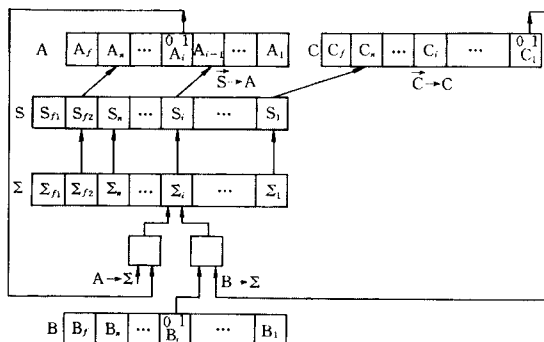


图2 定点原码乘法器的原理框图

作乘法时, 乘数放在乘数寄存器 C 中; 被乘数放在寄存器 B 中; A 中存放部分积和最终乘积的高位。 n 位乘法需作 n 步加法和 n 步移位操作。每一步乘法需根据乘数最低位 C_1 之值决定前一步部分积上是否加上被乘数, 然后将所得新部分积右移 1 位送寄存器 A 中, 其最低位移入乘数寄存器 C 的最高位, 与此同时, C 寄存器也右移 1 位, 以便根据乘数次低位之值 (此时该位已移入 C_1 中) 决定下一步乘法做什么。图 2 中 S 为移位器, 这样共作 n 步得到双倍字长的乘积, 高位乘积在 A 中, 低位乘积在 C 中。结果的符号由 $B_f \oplus C_f$ 得到。

上述定点运算器方案中的寄存器数目较少, 每次运算结果如果不为下条指令所用, 需写入主存储器中保存; 而当后续指令需要这个数据时, 又要从主存中读出, 增加了访问主存次数, 降低了机器的运算速度。因此, 提出了一种改进方案——通用寄存器

组方案,其中各个寄存器的地位都是平等的,作用都是相同的。寄存器的数量增加后,可以保存若干中间结果,减少访存次数。

浮点运算器 为了适应科学计算和工程设计的要求,扩大数据的表示范围,在计算机中一般需设置浮点运算器,在微型计算机中可作为选件,以满足不同用户的要求。浮点运算器分成阶码运算部分和尾数运算部分。阶码是定点整数,小数点的位置隐含指定在阶码数值最低位的右边;尾数是定点小数,其小数点的位置隐含指定在最高数值位的左边,符号位的右边。

浮点加法器原理框图如图3所示。

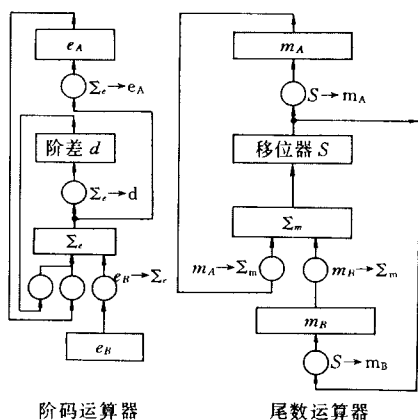


图3 浮点加法器原理框图

设两个规格化浮点数, $A = m_A \times 2^{e_A}$, $B = m_B \times 2^{e_B}$ 。A 放在 e_A, m_A 寄存器中; B 放在 e_B, m_B 寄存器中。浮点加法过程如下:

(1) 比较阶码 求阶差, $d = e_A - e_B$, 阶差存放在阶差寄存器 d 中。

(2) 对阶 小阶向大阶看齐, 调整阶码小的尾数。在 e_A 中保存大阶。

(3) 尾数求和 将两数的尾数 m_A, m_B 送入尾数加法器 Σ_m 求和, 结果送 m_A 中保存。

(4) 结果规格化 如果尾数最高数值位为非有效位, 需将尾数 m_A 左移 1 位, 结果的阶码 e_A 减 1, 直到结果的尾数为规格化数为止, 称为向左规格化, 简称左规。如果结果的尾数 $m_A \geq 1$, 则为尾数溢出, 需将尾数 m_A 右移 1 位, 阶码 e_A 加 1, 称为向右规格化, 简称右规。

(5) 舍入 结束浮点加法操作前需

进行舍入。办法有恒舍法、恒置 1 法、下舍上入法等。

为了提高运算速度, 在大型计算机的运算器中, 专门设置了乘法器、除法等。

为了表示每条指令运算结果的特征, 运算器中还设置了标志寄存器, 用来表示寄存器结果为零、结果溢出等状态。有些计算机把它们包括在程序状态字 PSW 中。

处理器总线及数据通路

在中央处理器中, 算术逻辑部件 ALU 是处理中心, 寄存器或存储器中的数据要送到 ALU 中进行处理, 处理结果又送到寄存器或存储器中去, 有时各寄存器之间的数据传送也可以 ALU 为中心进行, 这样可以节省许多连线, 其缺点是不少操作都要串行执行, 影响运行速度。这种方案的各寄存器都挂在 ALU 输入端上, ALU 为了选择参加运算的操作数, 通过多路开关与各寄存器相连。ALU 的输出端通过移位器送往各个寄存器。以 ALU 为中心的 CPU 数据传送通路原理图如图 4 所示。图中 $R_0 \sim R_3$ 为通用寄存器, 具有双端口输出, 用于存放操作数、中间结果, 也可作变址寄存器用。程序计数器 PC 可以利用 ALU 进行加 1 计算, 也可接受转移指令形成的有效地址。指令寄存器 IR 中地址部分 D 可以通过 ALU 进行变址等计算, 形成有效地址送存储器地址寄存器 MAR。存储器中读出的数据通过存储器缓冲寄存器 MBR 送入通用寄存器。多路开关下面的“与”门和“非”门用于实现逻辑运算。移位器可向各寄存器传送运算处理结果。

处理器总线可用作各个寄存器交换数据的公共

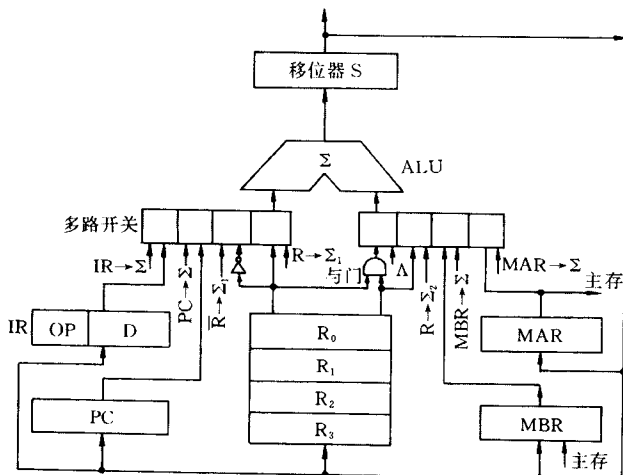


图4 以 ALU 为中心的 CPU 数据通路原理图

通路,处理器总线可以有一组或多组。图5为具有内总线的中央处理器原理图,它以单总线为例说明中央处理器的数据通路原理。图中的内总线作为所有寄存器传送数据的通路,简单而灵活,但只能分时使用,任何时候只允许1个寄存器占用总线,因此速度受到一定限制。取指令时,由 PC_{out} 控制把PC中的值送上内总线,再经过 MAR_{in} 将指令地址送入存储器地址寄存器 MAR , M_r 命令控制读该内存单元,并把读出内容送入存储器缓冲寄存器 MBR 。 MBR_{out} 把读出的指令送上内总线, IR_{in} 把指令取到指令寄存器 IR , IR 中的 s, d 分别表示指令中源(或目的)操作数地址存放的寄存器号,再经过 FAR 控制从通用寄存器 $R_0 \sim R_7$ 中取出源(或目的)操作数地址,送 MAR 。读出第一操作数放在 Y 中,第二操作数取到内总线上后即可执行 ALU 运算,结果放到 Z 寄存器中。需要把运算结果存入通用寄存器或内存时,再将 Z 中数据送上内总线,其他所需控制命令可以类推。

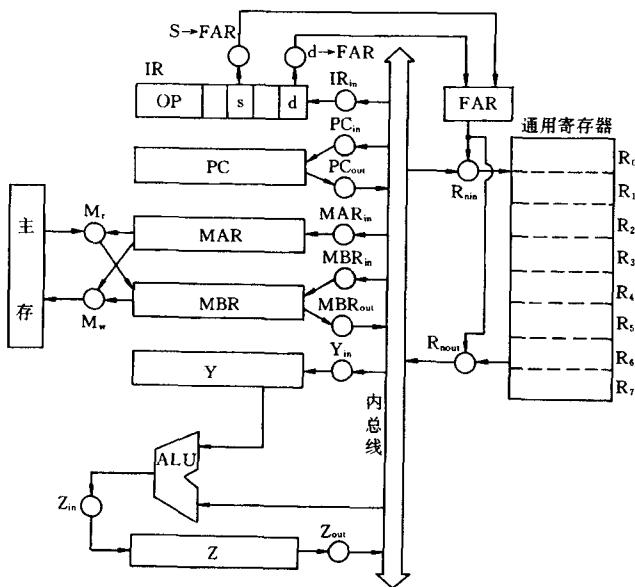


图5 具有内总线的中央处理器原理图

中央处理器的发展趋势

随着大规模集成电路技术的迅速发展,芯片集成度越来越高,中央处理器可以集成在一个半导体芯片上,称为微处理器,且功能不断增强。现代的中央处理器芯片包含整数运算部件、向量运算部件、地址部件、寄存器组、时序电路、指令堆栈、操作控制部

件、高速缓冲存储器、存储管理部件、总线接口部件等。甚至一个芯片上还集成了多个处理器。

参考文献

1. 金兰. 计算机组织与结构. 北京: 高等教育出版社, 1986
2. 王爱英. 计算机组成与结构(第三版). 北京: 清华大学出版社, 2001 (谢树煜)

zhongduan

终端 (terminal) 参见终端设备。

zhongduan fuwuqi

终端服务器 (terminal server) 用标准的串行接口将若干个终端连接到以太网或令牌环网上去的一种通信控制设备。图1是一个把许多终端通过两个终端服务器连接到以太网上去的结构图。

终端服务器终端侧的端口数,也就是它可带的终端数,一般为8或16,目前已可达32。

终端服务器所带的终端通常为异步终端,其传输速率可达几十kb/s。它也可以带个人计算机(处于仿真终端工作方式)、远程终端(通过调制解调器)或串行打印机等。

随着工业标准连网协议的增加,终端服务器应能支持多种协议。目前,有的终端服务器已能在同一以太网上支持Telnet, TCP/IP, LAT和SLIP等协议。(过介望)

zhongduan shebei

终端设备 (terminal device) 通

过通信线路或数据传输线路链接计算机的输入输出设备。简称终端。计算机终端设备通常设置在计算机中心以外更适合于该系统用户的地方。终端设备设置地点距计算机较远时,需在传输线路上加装调制解调器,这类终端称作远程终端。而终端与计算机距离较近,例如在同一建筑物内,不需经调制解调器传输的称作本地终端。

终端设备有许多类型,按用途可分为通用终端和专用终端。通用终端有交互终端与远程批处理终端两类。专用终端用于民航订票、旅馆房间预订、银

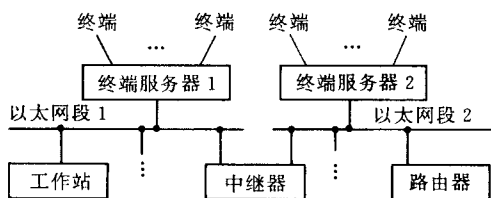


图 1 以太网上的终端服务器

行存取款、销售点收款记账、库存管理、教育、医疗以及其他方面,因业务的特殊性而有许多专门的类型,如销售点终端、银行窗口终端、自动取款终端、自学终端等。终端设备按功能完善的程度又可分为简易终端、灵巧终端和智能终端等。简易终端无缓冲能力,只能在低速异步传输方式工作,没有纠错能力。灵巧终端有数据缓冲能力,既可同步工作也可异步工作,有较高的传输速率,能检错和纠错,可对输入文本内容进行修改、编辑。智能终端备有微处理器、可编程序,既可联机运行,亦可独立工作,有较强的数据处理能力和文件管理能力。除上述终端设备类型外还有一些如小型便携式通用终端、光学字符阅读终端、汉字终端等。

交互终端 或称为分时终端,它可使计算机用户按分时操作与计算机对话,对每个单独请求作快速响应。每次向计算机发送一行请求、程序语句、数据。通常,交互终端的传输速率很低,因自终端发送的速率受到用户操作键盘速率的限制。但从计算机发至终端的信息则可以较快的速率传输。对智能终端、图形显示终端以及配备盒带驱动器及软磁盘驱动器的终端而言,则在两个方向均可以较快速率传输。通信链接往往是拨号电话式的。常见的终端设备类型有键盘打印终端与键盘显示终端两种。交互终端最早出现于 20 世纪 60 年代初,早期的终端设备主要是电传打字机。随着计算机交互应用的普及,许多专门设计的新型快速键盘印字设备陆续出现,性能显著提高,字符种类增多。显示终端明显优于打印终端。传输速率较快,一次可收发一行以上的电文,有些显示终端在发送到计算机之前可以修改所打的文本。显示部件因系电子部件,可靠性高。但显示终端不能留下永久的记录。为了弥补此缺点,通常在显示终端上配接印字设备。显示终端有字符显示终端及图形显示终端两种。

许多交互终端可接盒带驱动器、软磁盘驱动器存储设备。在不与计算机联机的状态下,用户可将输入内容记录到磁带或软磁盘上,而后,再以较快

速率将输入内容发送至计算机。同样,从计算机发来的输出亦可先记录在磁带或软磁盘上,然后再慢慢地由用户观察。

便携式通用终端主要由键盘、小型印字机及音响耦合器等三部分构成。当电话线路与计算机接通后,将话筒耳机压在音响耦合器上,终端设备即可工作。音响耦合器是在终端设备与电话线路间靠音响耦合的一种调制解调器。键盘输入信号经音响耦合器变换成音响信号传送给电话机之受话器,然后经电话线路送至计算机。反之,计算机之输出同样以音频方式传至用户电话机之送话器,经音响耦合至音响耦合器,输出正常电脉冲信号,驱动打印机工作。

远程批处理终端 可使用户像在计算机中心一样,远距离地使用计算机。远程批处理终端所配置的输入输出设备与直接连接在计算机上的相同。最简单的批处理终端可以有一台行式印刷机和一台卡片输入机。复杂一些的终端还可以有卡片穿孔机、磁带机、磁盘驱动器以及其他输入输出设备。事实上,小型(甚至中型)通用计算机常用作大型系统的批处理终端。传输线路可以是专线,也可以是拨号电话线,典型速率为 1 200 ~ 9 600 b/s,甚至更高。链接中需加调制解调器。简单的批处理终端不能同时收发,即只能半双向工作。而较复杂的批处理终端则可同时收发,即全双向工作。

专用终端 专用终端需专门的设计,以适合特殊业务的需要,便于操作人员尽可能方便和高效地完成其各自的工作。这类终端通常用专线连接,而且是多路复用,即多台终端设备共享一台控制器及专线。这类终端高度专门化,可以不包含通用键盘或打印机,而代之以特殊的指示灯、信用卡阅读器、光学扫描器等。在有些情况下,它们还可提供以记录的或合成的语音形式的输出。

银行终端 有安装在银行办理存款、汇兑及贷款等业务窗口的出纳员用键盘打印终端及自动存取款机终端等类型。

出纳员使用的终端的基本构成包括:微处理器为主体的控制器、连接通信线路的接口、存储器、键盘、专用打印机、指示板等,此外还可选接字符显示器、磁卡阅读器、磁条阅读器等。出纳员可依照显示器上显示的引导项目进行输入。存折可很方便地插入打印机的轨道中,自动到达打印的位置。终端并可读出存折磁条上的户主号码、存款余额、印字行数等信息。

由顾客操作的自动银行终端设备有自动取款机、自动存款机和自动存取款机(兼整钞换成零钞的功能)等三种类型,同时又有使用存折与不使用存折的区分。不使用存折的一种需使用磁卡。其基本结构包括:控制器、远程监视控制器、磁卡阅读器、存折插入、印字及退出机构、纸币分类保存箱与计数输出机构、纸币鉴别机构、纸币存入及不可分辨的纸币退还机构、结存便条打印及输出机构以及键盘指示器等。对三种类型的自动机器来说,不同之处在于取款机不需要纸币鉴别机构,而存款机又不需纸币计数输出机构。随着文字引导的显示还可伴随输出语音的解说。

教育训练终端 以 PLATO 计算机辅助教学系统中学生终端设备为例,在显示器屏上罩着一透明的特殊触摸键膜,学生利用这种触摸键可与计算机对话,跟随系统的引导进行各种作业的回答和进行各种实验。利用这种终端进行某些实验的训练还可免除危险,如化学反应及电力电网的实验等。

音频终端 由简单的语音输入设备和语音输出设备可构成音频终端。按键式电话机可作为一种音频输入终端,但键的种类仅限于 0 到 9 共 10 个数码。(刘锡刚)

zhu cunchuqi

主存储器 (main memory, MM) 计算机在运行程序时,存放按地址编址的指令与数据的存储设备。主存储器又称**内存储器**、**随机存取存储器**或**操作存储器**,它是存储系统的主要层次。主存储器的存取速度对整个计算机系统的运行速度有重大的影响。它的存取速度比辅助存储器的高很多,它的容量比高速缓冲存储器的大很多。

主存储器是按地址存取信息的。图 1 是主存储器的工作原理框图。它在工作时,中央处理器将地址送到地址寄存器,并发出“读出”或“写入”命令。地址译码器对地址进行译码,以确定相应的存储单元。如果是读出命令,则将存储单元的代码读出,送往代码缓冲寄存器;如果是写入命令,缓冲寄存器中的代码被写入存储单元。

由于主存储器的存取速度对于计算机系统的运行速度有着重大的影响,主存储器的实现技术一直受到重视。早期的主存储器曾采用过威廉斯管(一种特殊设计的阴极射线管)存储器和**水银延迟线存储器**。从 20 世纪 50 年代中到 60 年代末,主要采用**磁心存储器**,从 70 年代初以来都采用**半导体存储**

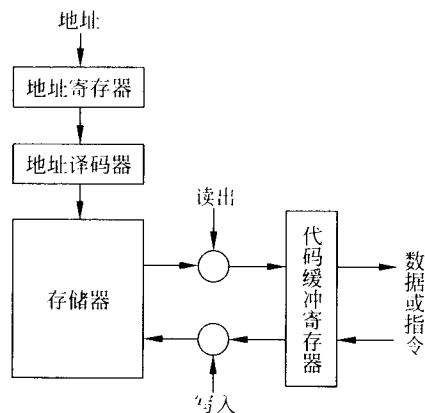


图 1 主存储器的工作原理框图

器。70 年代至今,主存储器的实现技术也从动态随机存储器 (DRAM)、同步动态随机存储器 (SDRAM) 发展到今天的双倍数据率的同步动态随机存储器 (DDR SDRAM) 和 Rambus 动态随机存储器 (RDRAM)。(郑衍衡)

zhuye

主页 (home page) 万维网 (WWW) 服务器上超文本置标语言 (HTML) 编写的文档的起始点和汇总点。主页是一种特殊的网页,它是用户开始浏览一个万维网文档的地方。用 HTML 编写的网页可包含文本、图像和指向其他万维网网页的指针。

在主页里,应给出一个万维网文档的概述,它所包含的主要内容,浏览各种信息的通道。使用户看到主页后,对这个万维网文档有一个概括的了解,对需要进一步了解的内容应该知道如何去查找。

每个主页有一个统一资源定位器 (URL), URL 描述了在超文本链接激活时所访问文档的地址。HTML 编写的文档中的超文本链接由两种方式构成:链接文本和链接图标。点击这些链接时,就可调用超文本,跳转到包含更多信息的某个资源。URL 还描述了访问目标服务器所用的协议、服务器名和文档所在的目录路径文件名。这种资源的统一命名规则使得万维网成为一个很好的信息集成环境,所以许多用户以万维网浏览器作为访问 Internet 的通用工具。

由于 HTML 文件是普通的 ASCII 文件,所以只要有一个字符编辑器就可以编写网页文件。但是,用普通的文本编辑器来编写 HTML 文件并不方便,

必须记住所有的 HTML 标记符号,书写起来非常繁琐,容易出错。现在,有一些工具有助于编写 HTML 文件。比如,有一种是可以直接编写 HTML 文件的工具,称为 HTML 编辑器;还有一种是将其他格式的文件转换成 HTML 文件的工具,称为 HTML 转换器。各种 HTML 编辑器和 HTML 转换器为编写 HTML 文件提供了非常简便的方法,使用户可以不必了解 HTML 本身就能够制作网页。

最简单的网页仅包括标题和正文两部分。在一个实际的网页中会使用多处链接,这种链接使得用户可以从当前阅读的文本迅速访问其他相关的文本信息。正是通过链接,我们才能把一张张独立的网页组成一个庞大的信息系统。在网页中可插入图像,可调整图像和文本的相对位置,可对图像进行修饰,建立图像链接以及用 gif 文件实现动画,这样使得网页更加生动,更有吸引力。还可以对屏幕进行横向或纵向的分割,使得在同一屏幕上出现多个窗口,每个窗口上显示不同的网页内容,并且可以独立变化。使用多窗口技术,可以使屏幕的信息量增大,使万维网网页更加吸引读者。HTML 还提供读者与万维网服务器进行交流的方式。例如,读者想查询某类信息,就把这类信息的查询条件通过浏览器输入到服务器中,万维网服务器根据读者输入的查询条件在服务器方进行查找,然后再把查询结果反馈给读者。对读者来说,这是一种交互式访问,同时也扩大了读者的浏览范围。这一交互方式是由 HTML 和驻留在万维网服务器上的程序共同完成的。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999

2. Douglas E Comer. Internetworking with TCP/IP, Vol. III. 3rd ed. Prentice Hall Inc., 1995

(胡道元)

zhuanjia xitong

专家系统 (expert system) 具有为解决特定问题所需专门领域知识的计算机程序系统。又称基于知识的系统。主要用它来模仿人类专家的思维活动,通过推理与判断求解问题。这个专门领域知识包括所设计的专家系统要解决特定领域问题的知识和用于求解这个特定领域问题过程的知识。一个专家系统主要由以下两部分组成:一个称为**知识库**的知识集合,它包括要处理问题的领域知识;和一个称

为**推理机**的程序模块,它包含一般问题求解过程所用的推理方法与控制策略的知识。推理是指从已有事实推出新事实(或结论)的过程。人类专家能够高效率求解复杂问题,除了因为他们拥有大量的专门知识外,还体现在他们选择知识和运用知识的能力方面。知识的运用方式称为推理方法,知识的选择过程称为控制策略。

由于专家系统的主要功能是模仿人类专家的智能活动,所以它还需要使用专家系统的用户与专家系统在求解问题过程中进行对话。这个人机交互的对话包括两个方面,一方面专家系统要像人类专家在诊断疾病或诊断机器故障时,与处理对象对话或通信一样,通过询问或观察获取更多信息;另一方面专家系统又要像人类专家一样,为用户解释它是如何求解问题的,或者推理过程中结论获得的理由,或者为什么所期望的结论没有达到的原因。

专家系统适用于缺乏合适算法求解问题而往往又能采用领域专家经验来求解问题的场合。经验性知识往往呈现不确定性或不精确性,仅以一定的可能性存在。此外,要求解问题本身所提供的信息往往也是不精确和不完备的。知识的这种不确定性,一方面是由于客观世界中有很多事物是表露不完全的,因而人们对它的认识也就不可能完全;另一方面,知识并不是只有真与伪这两种状态,还可能存在于许多中间状态。专家系统就是利用这些模糊的知识信息进行推理得出结论的。

构造专家系统的关键任务是获取知识、建造知识库。可以从人类专家处获取知识(如概念、事实和规则),特别着重于所处理特定问题领域中专家的经验知识;也可从现存的资源(书本等资料)中提取知识。

一个专家系统,除了知识库和推理机两个主要部分之外,还包括**知识获取**和人机界面这两部分。典型的专家系统组成如图1。

专家系统可用于求解不同类型的问题,这些问题可划分为若干基本类型,如表1所示。

专家系统的设计目前还没有通用的模式,有经验的知识工程师要先了解清楚求解问题的复杂性和可用的启发式求解问题的知识,再开始构造系统。系统的构造由以下四步组成:

(1) **知识获取** 逐步从专家、书本处获取问题领域的基本概念,即描述和求解问题的启发式知识,直至取得足以达到专家求解问题水平的知识。

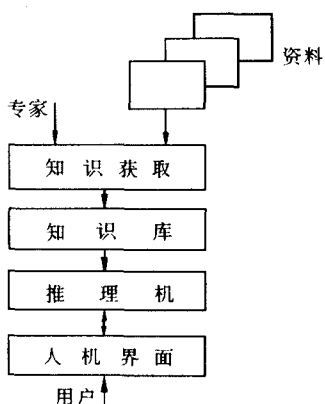


图1 典型的专家系统组成

表1 求解问题的基本类型

类别	求解的问题
解释	根据获取的数据对现象或情况作出解释
预测	在给定条件下推出可能的结果
诊断	从可观察的现象中推出系统的故障
设计	在限定条件下给出目标的设计
规划	设计一系列动作
监督	比较观察到的现象和期望的结果
排错	为故障确定补救的方法
控制	控制整个系统的行为
教学	诊断与纠正学生的学习和行为

(2) 知识表示 专家知识包含智能活动的能,但并没有包含表达这些知识的方法。因此知识工程师需将知识转换到可应用的形式,即选择合适的形式,表达求解问题的知识,实现工程化。

(3) 系统设计 设计系统的体系结构并进行知识程序设计,建造知识库和推理机。通常用专家系统开发工具进行建造。

(4) 系统精化 一般刚建立的专家系统性能较差,可能由于对专家求解问题的行为理解不正确,或者由于抽取的概念有误,或者由于表达知识不正确,或者由于忽视了某些细节。这类错误并不完全表示专家和知识工程师专业水平不够,因为在设计系统之前,专家并不具有有效方法去表达知识的经验。事实上,无缺陷的开发过程是不存在的。这就要通过实例检验,去直接反映出系统知识的脆弱性和缺陷,专家可以有针对性地改进知识库,逐步引导系统性能达到专家水平。

因此,专家系统构造过程一般有下列五个相互依赖、相互重叠的阶段:识别、概念化、形式化、实现与验证。五个阶段之间的关系如图2所示。

专家系统的实现一般是采用专家系统开发工具来进行的。在美国,绝大多数专家系统使用外壳这类开发工具实现,亦可使用程序设计语言实现。LISP语言是一种表处理语言,它是许多专家系统编程语言的基础。在欧洲和日本用逻辑编程实现专

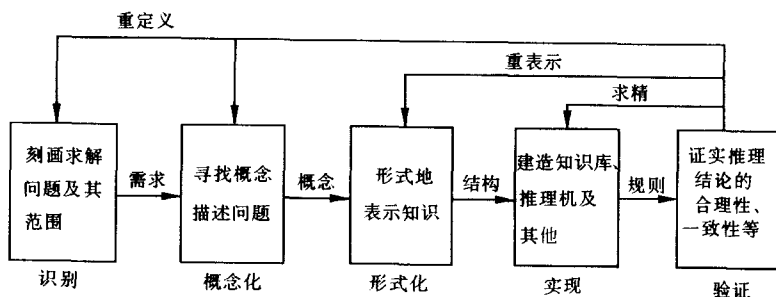


图2 专家系统典型的构造过程

家系统,广泛使用的语言是 PROLOG,它基于一阶谓词演算。PROLOG 程序是由一阶逻辑公理和要证明的定理组成的。

专家系统运行与维护都需要一个良好的支持环境。这个支持环境除包括前述的易学、易用的人机界面和根据需要能获取知识的工具之外,一般还要有清晰、易理解的解释工具和能方便地排除知识表

示中语法错误、语义错误的知识库编辑工具。

专家系统形成之前,人们一直在探索着如何用计算机模仿人的智能。1956年诞生“人工智能”这个术语之后,人工智能研究上取得两项重要成就,一是美国 Allen Newell, J. C. Shaw 和 Herbert A. Simon 合作编制的定理证明程序;另一个是 Arthur L. Samuel 的博弈程序。一般公认这两个成就是计算

机模仿人的高级思维活动的重大突破,是人工智能研究的开端。随后出现研究一般问题求解(简称GPS)的规律。在此基础上,1965年美国斯坦福大学Edward A. Feigenbaum首先与化学领域专家合作,研制了应用于有机化学分析的DENDRAL系统,其效果类同于专家。

从20世纪70年代后期以来,美国、欧洲、日本以及中国出现了一大批应用于各领域的专家系统,涉及医学、化学、生物、工程、法律、农业、商业、教育、军事等领域。工业界如美国的许多公司也投入了大量的开发力量。中国在“七五”计划期间,国家投资用于“实用专家系统”专项,其开发涉及工、农、医等领域,产生了很好的社会与经济效益。近几年来,在专家系统广泛应用于各领域的基础上,诞生了**分布式专家系统**和与其他信息系统相结合的新型综合的专家系统或智能信息系统(参见**连接专家系统**)。

专家系统已成为人工智能发展与应用的三大前沿之一(另外两个是**模式识别**和**智能机器人**)。它的成功标志着计算机处理对象从数值、数据处理阶段发展到了知识处理阶段,标志着计算机的应用在深度与广度上向前迈进了一大步,跨入了实现计算机智能化应用的新阶段。

参考文献

1. 林尧瑞,张钺,石纯一. 专家系统原理与实践. 北京:清华大学出版社,1988
2. Hayes-Roth F, Waterman D A, Lenat D B. Building Expert System. Reading MA: Addison-Wesley, 1983 (施鸿宝)

zhuanjia xitong kaifa huanjing

专家系统开发环境 (expert system development environment) 支持专家系统的开发、运行和维护的软件系统,主要由专家系统开发工具和专家系统支撑环境组成。

专家系统开发工具

它是辅助专家系统开发的程序系统,一般可分为专家系统外壳、专家系统开发语言和AI工具包三类。

(1) **专家系统外壳**是一个已实现的专家系统骨架。在外壳系统中,知识获取、知识表示、推理控制以及解释机制等都已固定,形成一个“空骨架”,开发者只要把获取的领域知识按外壳要求的知识表示形式填入知识库,即可形成一个面向具体领域的专家系统,这个过程大大缩短了专家系统的开发周期。

外壳的一般体系结构如图1所示。

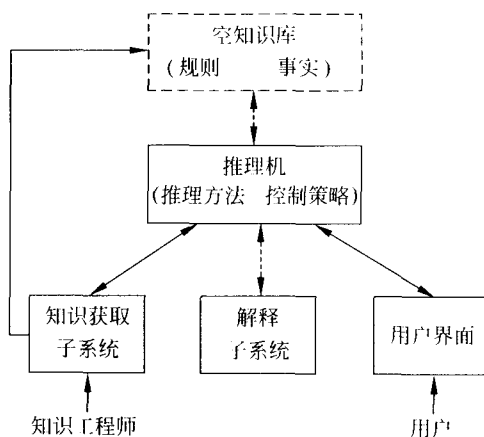


图1 一个典型的外壳系统结构

早期著名的外壳系统有EMYCIN和KAS。此后较为著名的外壳系统有AM, Crystal, Leonardo和EXSYS等。

脱胎于具体专家系统的外壳系统是一种高度专用的工具,所受限制较多,灵活性较差,往往仅适用于原专家系统求解问题的类型(如EMYCIN, KAS等)。在取得专家系统开发工具的经验后,专门开发出来的工具系统,如“天马”系统,OKPS系统等,由于不脱胎于任何具体的专家系统,因而所受限制较少,灵活性较好,对于问题求解类型的限制也较少。

(2) **专家系统开发语言**是供知识工程师建造专家系统使用的一类比外壳系统更灵活的支持知识处理的高级程序设计语言,也称知识处理语言。与常规的编程语言相比,该类语言在符号处理、模糊匹配、表示知识结构以及实施知识加工和基于知识的推理等方面均更具方便性和有效性。使用该类语言可以方便地表示知识,利用知识进行推理、求解问题,并可以将它嵌入某种宿主语言中,从而使知识库、推理执行机构和解释机构的建立以及用户接口的实现都更为灵活(与外壳系统相比)、方便和有效(与常规的编程语言相比)。

(3) **AI工具包**是除专家系统外壳和专家系统开发语言外,另一种专家系统开发工具。这些工具包包含专家系统的基本组成部件,如多种知识表示、多种推理机制和不同领域问题的求解模型。开发者通过工具包提供的选择机制针对具体领域问题灵活地选择各个部件并组装生成专家系统。AI工具包

具有比专家系统外壳更大的灵活性,能够开发多种类型的专家系统。比较著名的 AI 工具包有 Level 5 Object, ARTIM, KEE 和 LOOPS。

专家系统支撑环境

它是指辅助专家系统运行和维护的软件工具包,是帮助用户与专家系统的问题求解部分进行交互的程序,由程序设计辅助工具和系统功能增强工具两部分组成。前者包括辅助排错、知识库编辑等工具,后者包括用户界面、解释程序、知识获取等工具。

(1) 辅助排错工具 这是一种专家系统开发过程中排除错误的工具,一般包括跟踪工具、中断处理、测试工具等。跟踪工具为用户提供一个系统运行的轨迹。中断处理允许事先告诉程序在什么事件发生时停下来,以便用户能使程序在一些重发性错误或需要检查的程序状态出现时停止运行。测试工具能自动地检测系统,以找出系统的错误或推理结论的不一致性。

(2) 知识库编辑工具 这是为用户或知识工程师提供了一种修改知识库的工具,包括编辑工具、语法检查工具和语义检查工具。编辑工具在专家系统调试和运行过程中,提供对知识库进行增、删、改的手段和记录有关修改与被修改的知识信息;语法检查工具用以解释和纠正语法错误的知识文本;语义检查工具检查输入知识的语义,以发现与已有知识的冲突,并给出解释以及冲突解决办法。

(3) 用户界面 提供一种系统运行时通过人机对话获取有关信息的方式。一般专家系统需用智能询问的方式,即在运行过程中由系统主动地、有针对性地提出问题或做出解释。这种相互间的提问和回答,可以是简单的是(yes)、非(no)或未知(unknown);也可是选项方式;也可用约定区间内的数字来表示不同程度,由系统按线性函数或隶属度函数等进行处理;还可用一定范围内的受限自然语言方式对话,或用图形、图像方式显示。目标是使人机对话方式更自然、更方便、更容易理解、更具有智能性。

(4) 解释程序 当用户对专家系统的推理提出疑问时,专家系统给出的回答机制。基本要求是:①能解释有关系统的知识和行为方面的问题;②对用户的提问能给出一个正确的和易于理解的回答;③使用户易于使用。

(5) 知识获取工具 是帮助知识工程师获取专家知识的程序。知识工程师在构造或扩充专家系统

的知识库时,需要通过某种方式,从某领域专家、实例(数据)或书本资料等知识源中获取知识,以某种知识表示方式存入知识库中。一般习惯上将人工和半自动知识获取方式纳入专家系统支撑环境,而自动知识获取已成为一个重要的独立研究领域(参见知识获取)。

参考文献

1. Joseph Giarratano, Gray Riley. Expert Systems-Principles and Programming (Third Edition). PWS Publishing Company, Boston, USA, 1998
2. George F Luger, William A Stubblefield. Artificial Intelligence Structures and Strategies for Complex Problem Solving (Third Edition). Addison Wesley Longman, Inc., 1998
3. Keith Darlington. The Essence of Expert Systems. Prentice Hall, 2000

(刘大有)

zhuan yong luo ji ji cheng dian lu

专用逻辑集成电路(application specific logic integrated circuit, ASLIC) 按照用户的特定要求而专门设计制造的一类数字逻辑集成电路。它是**专用集成电路(ASIC)**的一个重要分支。使用专用集成电路可使电子系统具有独特的功能和性能,且不易被仿制。同时由于设计方法的进步与设计工具的完善,其开发周期已大为缩短,开发成本也大幅度降低。

按设计方法不同专用集成电路可分为三种:①定制电路,如标准单元电路、全定制电路;②半定制电路,如门阵列(GA);③可编程逻辑电路,如可编程逻辑器件(PLD)、现场可编程门阵列(FPGA)。

专用集成电路的开发方法取决于用户对性能、费用和开发周期等需求的综合考虑。定制电路一般开发周期略长,但性能最好,当生产批量大时性能价格比最优。门阵列开发周期较短,适用于中等用量的用户。可编程逻辑电路开发周期最短,适用于需要量较少的用户。

专用集成电路广泛用于计算机及各种外部设备,同时也广泛应用于通信、工业控制、武器装备以及消费类电子产品(如洗衣机、电冰箱、电视机、照相机)中。

标准单元逻辑电路 采用标准单元方法设计制造的一类专用集成电路。标准单元是与数字逻辑电路中的各种门电路、触发器、寄存器、加法器等相对应的集成电路设计单元。一个标准单元具有三种形

态,即用于逻辑模拟的逻辑符号(含功能描述),用于性能验证的延时参数以及用于版图布局布线的单元版图。延时参数与专用集成电路制造厂家的工艺参数、电学参数密切相关。数以百计的这种不同的标准单元就构成了一个完整的标准单元库。

根据用户的逻辑图,设计工作站即可从标准单元库中调用与之对应的标准单元进行逻辑模拟和验证。逻辑图获得认证后,即可用标准单元进行布局布线,形成用来控制电路制造的完整版图。通过工艺加工、封装、测试,完成标准单元逻辑电路的制造工作。

由于标准单元通常都经过优化设计,在布局布线版图中无冗余单元,因而性能较好,占用的芯片面积也较小。虽然开发成本略高,但当生产批量较大时,采用标准单元法开发专用集成电路仍是最佳选择。

标准单元逻辑电路在计算机、通信等电子系统中应用广泛,有很大发展前景。

门阵列 利用预先制作的由大量器件规则排列成阵列的基片,按照用户要求进行连线封装等加工制成的专用集成电路。门阵列电路包含基片、宏单元和布局布线三个组成部分。基片是用基本单元按照一定规则排列而成的阵列,基本单元则是未经连接的一组晶体管。每个宏单元对应于一组特定的连接线,它可以将一个或几个基本单元中的晶体管连接起来构成具有一定功能的逻辑电路,如门电路、触发器、加法器。布局布线的作用就是按照用户逻辑图的要求,确定宏单元的种类、数量及它们在基片上占有的位置,并确定这些宏单元之间及它们与芯片的输入输出管脚之间的连接。

门阵列基片独立于用户的特定需求,它可由专用集成电路制造厂家批量预先制作。应用时,根据用户的逻辑图,仅需在工作站上模拟验证,通过布局布线形成包含宏单元及其相互连接的文件,并以此为基础制作出互连掩模,按照这些掩模在基片上形成连接线,经过封装、测试,即可完成专用集成电路的制作。

门阵列电路具有开发周期较短(一般为4至6周)、开发成本较低的显著优点,在计算机、通信等电子系统中普遍使用。

现场可编程门阵列 一种可由用户通过现场编程实现特定功能,面向多用户的专用集成电路。用户可根据其电路设计要求,选购相应集成规模、性能的FPGA产品,利用专门的开发系统,经过模拟验证,布局布线,生成编程文件,对选购的FPGA进行

编程,以开发成用户需要的专用集成电路。

与PLD不同,FPGA的核心部件是可编程逻辑功能块阵列与可编程连接线网。每一逻辑功能块经过编程可实现数十种不同的逻辑功能。连接线网经过编程可以完成逻辑功能块及输入输出电路相互间的各种连接。

编程由编程元件阵列及编程控制电路来实现。编程实质上就是根据用户的逻辑设计,确定逻辑功能块内部的连接线及它们之间的连接线网的通断状态。因此编程元件应具备开关及状态存储两种功能。常用的编程元件有静态随机存取存储器(SRAM)单元、熔通器、紫外光可擦编程只读存储器(UV EPROM)、电可擦编程只读存储器(EEPROM)以及快可擦编程只读存储器(Flash EPROM)等。采用SRAM编程的缺点是程序将因停电而消失,因此需要有可编程只读存储器(PROM)存放程序。熔通器的缺点是仅可实现一次性永久编程,采用UV EPROM或EEPROM则可重复编程且程序不会因停电而消失。

FPGA的优点是开发周期短,一次性开发费用低。其不足之处是芯片利用率低,编程元件的延迟影响了电路性能,器件价格较高。因此,FPGA一般用于电子系统的样品开发,适于小批量生产。

可编程逻辑器件 可由用户通过编程实现特定逻辑功能,具有一定通用性的专用集成电路。用户可按照自己的设计要求,利用专门的开发工具,对PLD进行编程以构成专用的逻辑集成电路。

PLD由编程元件阵列与编程电路、寄存器、输入输出电路等组成。它的逻辑功能主要由产生乘积项的可编程“与”门阵列与产生和项的固定的“或”门阵列来实现。PLD内部的编程元件可以是熔丝,或紫外光可擦编程只读存储器,或电可擦编程只读存储器。采用熔丝作为编程元件的PLD只能一次性编程,不能改写。采用UV EPROM作为编程元件的PLD其封装带有透明窗口,以使用紫外光擦除,再用电气方法改写,从而允许多次编程。采用UV EEPROM作为编程元件的PLD可用电气擦除和改写,允许大量重复编程。

PLD的开发周期短,广泛用于电子电路的验证,适于小批量生产。

通用逻辑阵列电路(GAL)是可编程逻辑器件的一种,其内部结构基本上与PLD的相同。它采用EEPROM作为编程元件,允许多次重复编程。

(仇玉林)

zhuanyong xiao jiaohuan ji

专用小交换机 (private branch exchange, PBX) 某个单位所拥有的或租用的电话交换机。它把内部电话互连起来,内部电话通过它可访问公用电话网,并可进行数据交换。

最初的专用小交换机(PBX)是人工操作的交换台,在一块开关板上由人工完成所有的连接。在20世纪20年代以后,被专用自动小交换机(PABX)所代替。PABX不需操作员干预,只要用户拨号便可自动接通电话。第一代PBX采用机电技术和模拟信号。用户间要进行数据交换时,必须通过终端、电话和调制解调器。20世纪70年代中期,推出了第二代PBX,它采用的是电子技术而不是电磁技术,内部的交换是数字化的,这种系统称为数字PBX或计算机化小交换机(CBX)。终端用户访问外部计算机时,不再需要用电话和调制解调器。第三代PBX是综合声音和数据的交换系统。

数字PBX的优点:

(1) 由于用数字方式处理所有内部信号,使PBX便于使用廉价的大规模集成电路和超大规模集成电路(参见集成电路)。

(2) 数字信号便于使用时分多路复用(TDM)技术,能有效地利用内部数据通道和访问公共TDM载波设备。

(3) 由于控制信号本身也是数字形式的,很容易通过TDM把它们集中到数字传送通道,产生信号的设备与传输介质无关。

(4) 数字信道更容易采用加密技术。

专用小交换机的一般结构 图1给出了数字PBX的一般结构。其核心部件是数字开关网络。它负责对数字信号流进行操作和交换。数字开关网络可由几级空分和时分开关组成,或简单地用时分多路复用开关组成。接到开关的是一组接口单元,通过接口单元与外界进行相互访问。通常接口单元完成同步时分多路复用功能,以适应多条输入线路。为了达到全双工操作(参见串行传输),每个接口单元要用两条线与开关相连。在输入时,接口单元执行多路复用操作。每条输入线按规定的速率采样,输入数据进入缓冲区,按时间片组织数据块,并按控制单元规定的时序,以PBX的内部速率(50~500 Mb/s)逐块送给开关。在输出时,接口单元在指定的时间片内接收来自开关的数据,经过多路复用缓冲后发送到相应的输出口。如图1所示,接口单元有用于处理数据、语音和中继等3种作用。

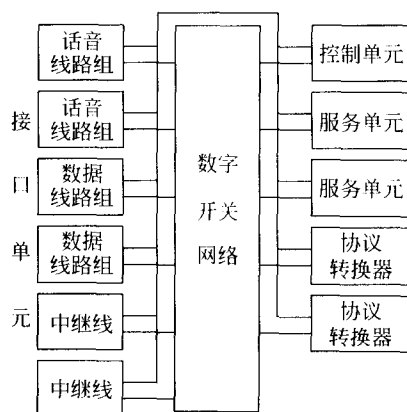


图1 专用小交换机的一般结构

控制单元的作用是控制交换开关,并与连接的设备交换信息。服务单元含有音调发生器、忙音发生器、拨号寄存器等。协议转换器用于对不同类型的线路进行适配,每条线路都与协议转换器相连接。

上述PBX的一般结构具有高可靠性,任何一个接口单元的故障只影响少数几条线路。关键部件,例如控制单元,可以做成有冗余的。为了可靠和经济,较大的PBX往往采用分布式结构。它由中央开关和1个或多个分布式模块组成,其间用同轴电缆或光缆连接。

专用小交换机在数据交换中的应用 一个配置完善的数字PBX可提供多种数据交换功能:

(1) 可将终端和主计算机连接在局域网环境中,即可将大量分布的终端、个人计算机和 workstation 通过PBX连接到一些主计算机上,以便进行数据通信;

(2) 将用户计算机通过公用交换电话网或租用线路网连接到远程计算机,进行数据通信;

(3) 若PBX配置了协议转换器或网间连接器,则可进行网络互连,例如在PBX中,若配置了X.25网间连接器,则可使终端通过公用分组交换网(参见公用数据网)与远方的X.25主机进行通信;

(4) 如果PBX中配置有综合语音(言语)、数据交换设备、传输设备及相应软件,甚至图像处理设备及软件,则PBX可以提供语音(言语)、数据、图像的综合服务。

参考文献

1. Stallings W. Local and Metropolitan Area Net-

work. Fourth Edition. New York: MacMillan Publishing Company, 1993

2. 胡道元主编. 计算机网络工程指南. 北京: 电子工业出版社, 1993 (史美林)

zhuanhuan jiance huanchongqi

转换检测缓冲器 (translation lookaside buffer, TLB) 在虚拟存储器系统中实现虚实地址转换的硬件机制。它是专门用于页表或段表(参见**虚拟存储器**)的高速缓冲存储器。现以页式虚拟存储器为例加以说明。由于页表较大, 只能放在主存中。许多系统的虚地址空间极大, 甚至简单的页表都不能在主存中放下。以 32 位逻辑地址为例, 若页长 1 024 B, 则页表中共有 $2^{32} \approx 4 \times 10^6$ 项。这时, 或者在内存中只放一部分页表(另一部分存在属于虚空间的磁盘上), 或者采用多级页表方案。总之无论什么方式都只会加长访存的延迟, 因此, 硬件的设置是必不可少的。

正如高速缓冲存储器和虚拟存储器的工作都是基于程序的访存局部性, 页表的访问也存在着局部性。因此, 一种简单的方式就是让页表项进入数据高速缓冲存储器, 以节省地址转换的周期数。但即使这样, 有效地址也需要 1 个周期后才能得到。另一种简单的方式是设置 1 个专门的寄存器, 用它记住上次转换的虚页号和实页号, 从而当本次访问针对同一页时, 不必再进行转换。

转换寄存器的更一般形式是由硬件提供专门用于页表项的高速缓冲存储器, 称为 TLB。有时也称为地址转换高速缓冲存储器(ATC)或目录检测表(DLAT)。TLB 的工作机制同高速缓冲存储器完全相同。因为 TLB 相对较小(通常含 64 项 ~ 1 024 项), 所以其访问时间极短, 从而只要 TLB 命中, 实地址就可以在 1 个周期内得到。如果 TLB 不命中, 则地址转换仍需经过内存中的页表(也许访问页表还会导致缺页), 且 TLB 的内容也必须更新。所幸的是, TLB 的命中率极高, 一般都超过 99%。当 TLB 不命中时, 经常能在数据高速缓冲存储器中找到所要的页表项(PTE)。

鉴于 TLB 对虚拟存储器系统性能的影响, 80 年代以来所有的虚拟存储器系统都采用了 TLB, 尽管它们中有些不提供数据高速缓冲存储器。为了进一步减少进程上下文切换时清除 TLB 内容带来的性能损失, 一些计算机系统还为系统态和用户态各设置了 1 个 TLB。

参考文献

郑纬民等. 计算机系统结构. 北京: 清华大学出版社, 1992 (唐志敏)

zhuangru chengxu

装入程序 (loader) 把保存在外存介质上的代码(通常是目标程序)装入内存并启动执行的程序。

根据目标程序的形式, 装入程序可分为三类: 绝对装入程序、再定位装入程序和连接装入程序。如果待装入的目标程序中的所有地址都是绝对地址, 则装入过程是根据装入地址、启动地址及目标程序大小等信息, 将目标程序读写到指定的内存位置, 从指定的启动地址开始执行, 完成这种过程的程序称为**绝对装入程序**。如果待装入的目标程序中的地址是相对地址, 则可将目标程序置于内存的任何位置, 装入程序根据存取方式的不同调整目标程序中的地址, 这种装入程序称为**浮动装入程序**。**连接装入程序**结合了浮动装入程序和**连接编辑程序**的功能, 它把许多分别编译后的目标程序组合成一个可执行程序, 再装入内存并启动执行。

装入程序可以是一个独立的程序, 也可以是操作系统的一个部分。装入程序本身常驻留在内存, 它的装入是由引导程序完成的。

一般装入工作是在整个程序执行前完成, 但是有些情况下需要在程序运行中再装入某些程序。可在程序执行中装入新的目标程序并将控制传递给该程序的装入程序称为**动态装入程序**。

参考文献

Beck L. L. System Software: An Introduction to Systems Programming. 2nd ed. Addison-Wesley, 1990

(张素琴)

zhuangtai zhuanxitu

状态转移图 (state transition diagram) 一种表示系统或系统元素的状态及其转移的图形工具。又称状态图。其基本构造包括状态、事件和状态转换。状态是由一组属性定义的, 表示系统或系统元素所处的阶段、可实施的活动以及可见的特征; 事件是由激发状态发生变迁的条件与活动定义的, 通常是由消息规约的; 状态转换是状态之间的一种关系, 例如顺序关系、并发关系等。

状态转移图是一个有向图, 图中的结点表示系

统或系统元素的状态,有向边表示状态转换,可以在每条边上给出引发状态转换的事件。

在实践中,可以采用状态转移图来表达系统的行为模型。为了控制表达系统或系统元素动态行为的复杂性,可以采用分层的状态转移图。

(李宣东 王立福)

ziyuan gongxiang moshi

资源共享模式 (resource - sharing mode)

20 世纪 80 年代伴随个人计算机和局域网的出现而产生的一种网络计算模式。在资源共享模式中,用户的应用程序和数据保存在文件服务器上。应用程序运行时服务器把文件从文件服务器下载到用户的桌面计算环境中,用户的作业在桌面环境中运行。

资源共享模式适用于文件的共享程度不高,并且要传输的数据量不大的应用环境。20 世纪 90 年代以后,由于并发用户数量的增加和应用程序急剧膨胀,资源共享模式已经逐渐被客户-服务器模式替代(参见客户-服务器计算)。

资源共享模式的出现,在某种程度上是由于早期的计算资源(例如硬盘等)的价格昂贵所致。资源共享模式的优点就是能最大程度地共享资源。

资源共享模式的缺点是要传输的数据量很大,对网络带宽要求较高。

(王勇)

zichengxu

子程序 (subprogram) 与子计算任务相应的处理对象和处理规则的描述。它是一个可被其他程序(单位)调用的程序单位。例如过程,函数,子例程。

在程序设计语言发展的早期,已认识到有些程序要在不同位置多次使用。如对数函数,各类三角函数等。为方便起见,就要求此类函数一经程序人员编码,就可在任何所需的地方使用。

子程序包括定义和调用两个方面。前者是定义子程序算法,后者是子程序的使用。下面用一个 ALGOL 语言程序的例子来说明过程(即子程序)的定义与调用。

```
begin
  array B[1:100,1:100], C[1:50,1:50];
  procedure TRANS(A,N);
```

```
  array A; integer N; value N;
  begin integer I,J; real TEMP;
    for I:=1 step 1 until N do
      for J:=1 step 1 until N do
        begin
          TEMP:=A[I,J];
          A[I,J]:=A[J,I];
          A[J,I]:=TEMP
        end
      end;
    .....
  过程调用          TRANS(B,100);
  .....
  过程调用          TRANS(C,50);
  .....
end
```

过程 TRANS 定义为求一个 $N \times N$ 矩阵的转置矩阵。在 ALGOL 等语言中,矩阵称为数组。执行过程语句 TRANS(B,100) 将 B 转置,结果仍在 B 中。在 ALGOL 语言中,若在 Procedure 前冠以类型名,则此过程为函数过程,此函数过程中应有对此过程名的赋值。函数过程调用(ALGOL 60 中称为“函数命名符”)可出现在表达式中,这与通常所说的过程不同。

在 FORTRAN 语言中不称过程,而称子程序。FORTRAN 子程序可分为两类,即 SUBROUTINE 子程序和 FUNCTION 子程序。函数(子程序)的结果值在函数体中必须被赋给它的名。而子例程(子程序)则否。它只能通过 CALL 语句,如 CALL TRANS(B,100) 来调用。不能在赋值语句的表达式中调用。

在 Ada 语言中,子程序定义为过程或函数。除在书写格式上稍有差异外,Ada 过程本质上类似于 ALGOL 过程,函数则类似于 FORTRAN 的函数子程序。Ada 还允许参数默认,即在调用时可不指定相应的实在参数,而实际值已事先在子程序声明中规定了。

C 语言程序由程序员定义的若干函数构成。C 不提供类似 PASCAL, ALGOL 语言中过程这种语言单位。然而,可以在 C 的函数体内把控制转回到调用函数处而不回送函数值,这一点很像“过程”;当回送函数值时它是一个函数。对于具有回送值的函数,调用程序可以使用这个值,也可以不使用这个值。如果在定义 C 语言函数时,在函数名前冠以

void, 则表明该函数无回送值。

C++ 和 Ada 语言都允许函数名一名多用。即同一个“函数名”可以有不同的含义。

参考文献

1. 徐家福. 系统程序设计语言. 北京: 科学出版社, 1983
2. 郑国梁, 钱士钧. Pascal 语言. 北京: 中国铁道出版社, 1989
3. 孙玉方, 文强. C 语言. 北京: 中国铁道出版社, 1989 (段祥)

zilicheng

子例程 (subroutine) 可用于一个或多个计算机程序中, 也可用于一个计算机程序的一处或多处的子程序。

1951 年, M. V. Wilkes 等人提出子例程思想, 目的在于借以将复杂的任务分解成若干较小的单位, 然后分别处理每一单位。

子例程肩负程序执行时必要的特定任务。因此, 它是程序的一部分, 甚至可以作为另一个子例程的一部分。鉴于同一个子例程可在程序中多次被利用, 这便节省了程序设计时间和代码所占有的存储空间。虽然子例程在高级语言之外有其广泛的意义, 但在高级语言这一范畴内, 术语子例程与子程序可以通用。例如, 在 ALGOL 语言中, 子例程是作为过程来实现的; 在 FORTRAN 语言中, 它是作为 SUBROUTINE 子程序和 FUNCTION 子程序来实现的; 在 C 语言中, 它是作为函数来实现的; 而在汇编语言中, 它是作为相关宏指令设施而被公用。

子例程分两类: 开式子例程与闭式子例程。开式子例程在调用处嵌入相应的子例程定义; 闭式子例程则在调用时直接转至相应的定义, 执行后返回。图 1 中给出了闭、开式子例程的使用说明。

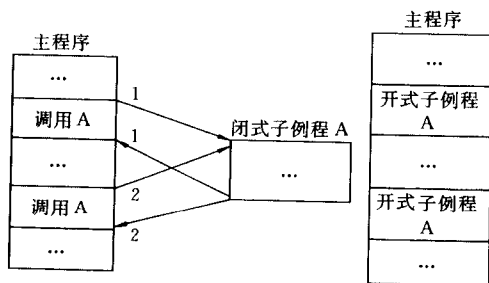


图 1 闭、开子例程

编译程序目标代码优化的一种技术便是将简短的子例程代码嵌入到调用它的地方, 借以显著节省机器执行时间并使存储器占用空间增加不多。RISC 计算机编译程序是采用这种技术的一个范例。

例如, FORTRAN 函数 ABS(X) 的值为变元 X 的绝对值, 编译程序通常按开式子例程实现, 这是因为它只需要两条机器语言指令便可完成:

CADDL(X) 清累加器, 将 X 单元的内容送到累加器

SSP 置累加器符号为正

如果 ABS(X) 按闭式子例程实现, 仅用于进、出子例程的机器时间就比执行上述两条指令所花的时间要多得多。特别是对 RISC 计算机情况更为严重。至于哪些函数和子例程作为开式子例程使用完全由它对存储空间和执行时间的要求这对相互矛盾的因素来决定。子例程有可复用子例程, 可再入子例程, 递归子例程等(参见例程)。

参考文献

1. Wilkes M V, Wheeler DJ, Gill S. The Preparation of Programs for an Electronic Digital Computer. Reading: Addison-Wesley, 1951
2. Barnes J G P 著. Ada 程序设计. 张乃孝等译. 杨英清等校. 北京: 人民邮电出版社, 1989 (段祥)

zifuji

字符集 (character set) 按某种约定而设定的一组表示数据的符号。数据可以是数、英文字母、汉字、符号、命令、图形、图像、声音等。在字符集内, 每个字符都有确定的二进制编码, 并能被计算机识别。

在编码字符集中的字符代码必须具有: ①惟一性, 即字符与二进制代码之间为一一对应关系。不存在 1 个字符有 1 种以上的代码, 或 1 种代码表示 1 种以上的字符。②规范性, 各种编码字符集都有相当大的适用范围, 在此范围内的计算机用户必须严格遵守其规定。只有这样, 才能保证信息的交换和相互利用。③兼容性, 当一个计算机系统中采用 1 种以上的字符集时, 应考虑不同字符集中字符代码的兼容性。即一方面, 要使各种字符共容于一个系统中, 能够相互区分而不会混淆, 另一方面, 也要考虑不同字符集之间具有某种共性和继承性而不会相互冲突。

计算机中应用最广的编码字符集为美国制订的美国信息交换标准码 (ASCII)。它采用 7 位二进制位进行编码。字符集中包含 32 个控制字符和 96 个

图形字符,图形字符有数字、英文大小写字母和多种符号。与 ASCII 完全兼容的字符集有国际标准化组织制订的 ISO-646 及我国制订的 GB-1988。国外比较流行的另一种字符集为扩充的二-十进制交换码 (EBCDIC)。这是一种以 8 位二进制位编码的字符集,主要应用于美国 IBM 公司的大中型计算机系统中。为了适应汉字信息处理的需要,我国于 1980 年制订了“信息交换用汉字编码字符集·基本集”,其标准号为 GB 2312—80,简称国标码。它是我国应用最广的汉字编码字符集。GB 2312 包含汉字 6 763 个,非汉字图形字符 682 个。每个字符以两个字节来编码。当在计算机中同时处理汉字及 ASCII 字符集的字符时,为了满足兼容性的要求,将 GB 2312 的代码作某种变化,即把字节的最高位设置为 1,以便同最高位为 0 的 ASCII 码相区分。为了提高字符集的覆盖能力,我国还制定了 GB 7589, GB 7590, GB 12345 等辅助汉字集。其中 GB 12345 主要用于繁体字。

除了 GB 2312 外,我国台湾地区还采用“通用汉字标准交换码(即 CNS 11643)”、BIG5 码及 TCA 码等。随着汉字在国际交流中的重要性日益提高,在国际标准化组织以及中、日、韩 3 国专家和政府的努力下,制定了包含中、日、韩 3 国所使用的汉字的编码字符集“统一的中日韩汉字辞汇与字序”,简称“CJK”大字符集。该字符集包含 20 902 个汉字,已成为“国际标准通用多八位编码字符集 (ISO 10646)”的重要组成部分。我国所制定的 GB 13000—1993 字符集标准与之完全兼容。由于 ISO 10646 标准的完全实现比较困难。我国在 2000 年公布 GB 18030—2000 标准,该标准字符集中包括 27 484 个汉字,完全覆盖并替代一度作为 GB 2312 标准重要补充的《汉字扩展内码规范》(即 GBK)。GB 18030—2000 标准作为国家强制性标准。

(黄震春)

zibianyi chengxu

自编译程序 (self-compiler) 用被编译的语言自身来书写的编译程序。

一般说来,任一编译程序都涉及到三个语言:源语言 S,目标语言 T 和实现语言 I。源语言是被该编译程序编译的语言;目标语言是书写目标程序的语言;实现语言是用来书写该编译程序的语言。可以用注明了这三个语言的 T 图 (图 1) 来表示一个编译程序,或写为 $S_I T$ 。例如,用 C 语言写的目标语

言为 80386 宏汇编的 C++ 编译程序可表示为图 2,或写为 $C_{++C} 80386$ 。

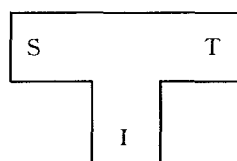


图 1 编译程序的 T 图

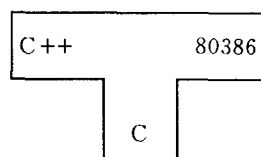


图 2 T 图一例

自编译程序的最大特点是用源语言本身作为实现语言,它的 T 图呈图 3 形式。

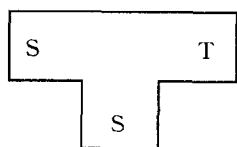


图 3 自编译程序的 T 图

因为语言 S 的编译程序 $S_S T$ 是用 S 写的,因而也是 S 的一个程序,所以该编译程序 $S_S T$ 需要由 S 的另一个编译程序 $S_{I_0} T_0$ 来编译。用 $S_{I_0} T_0$ 对 $S_S T$ 进行编译,得到的结果是用 T_0 写的目标语言为 T 的 S 的编译程序 $S_{T_0} T$ 。这一编译过程可用如图 4 的 T 图来表示。这种编译过程称为自展。

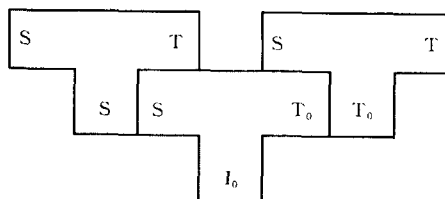


图 4 自展

自展有许多用处,例如,可以从一个语言的较小的子集自展出该语言的编译程序;还可以从一个语言的未优化的编译程序构造出该语言的优化编译程序。

参考文献

- Aho A V, Sethi R, Ullman J D. Compilers Principles, Techniques, and Tools. Addison-Wesley, 1986
(徐永森)

zidi xiangshang fangfa

自底向上方法 (bottom up method) 一类软件开发方法, 首先从系统实现的最基础部分着手, 由简单到复杂, 逐层向上构造, 直到最后得到所要的软件系统。

以程序设计的情形为例。先实现最基本的子程序、函数, 在此基础上构造出功能更强的函数来。每一个新的子程序, 或者直接实现, 或者调用已经实现的(比它更低层的)子程序来实现。直到最后实现整个系统。方法的特点是: 在开发的每一步, 系统的开发者手中总是有功能越来越强的子程序可供使用, 但是只在最后一步才得到整个系统(也许只需要调用很少几个高层子程序就实现了)。相应的测试工作也可以是自底向上的, 即按照构造的同样次序。或者边设计边测试, 即对设计好的功能、部件、子程序, 立即着手测试。这样, 开发者手中随时总是有可信赖的半成品。

代码级的软件复用可以看成是自底向上方法, 这时系统的基础部分是可复用构件(参见**软件复用**)。

自底向上方法也可能和其他方法结合, 即用其他方法设计(如**自顶向下方法**), 而用自底向上方法于实现。在文化程序设计系统中, 则可以随意地混合使用各种设计策略, 包括自底向上在内(参见**文化程序设计**)。

参考文献

- Sodhi J. Software Engineering: Methods, Management, and Case Tools. Blue Ridge Summit: TAB Professional and Reference Books, 1991
(董毓美)

ziding xiangxia fangfa

自顶向下方法 (top-down method) 一类开发软件的方法, 这类方法强调开发过程是由问题到解答、由总体到局部、由一般到具体。自顶向下方法提供了从高层次(问题定义)到低层次(编程实现)的分层分解的决策策略和决策方式, 并在具体的开发方法中给出相应的描述工具和设计步骤。自顶向下方法体现了**逐步求精**和信息隐藏的原则。所谓逐

步求精即从最能直接反映问题的基本概念和总体结构出发, 逐步精化、具体化、补足细节, 直到成为可以在机器上执行的程序。逐步求精离不开信息隐藏, 在精化的每一层上, 把其组成部分的详细内容隐藏起来, 留给下层来设计。所以逐步求精使得软件具有较好的层次结构。

发展简史

20 世纪 60 年代中期, 在大型软件系统的开发中遇到“软件危机”, 促使人们去考察软件设计本身固有的问题。自顶向下方法就是在这背景下, 与结构程序设计及软件工程同时产生的, 并一直伴随和促进着软件工程的发展。

1968 年, E. D. Dijkstra 首先指出: 为了得到正确的程序, 应在编程时采用适当的构造性方法, 并指出在程序中使用 GOTO 语句是有害的。他的意见引起了人们对程序设计方法的普遍重视。由此产生了以自顶向下逐步求精为核心内容的结构程序设计方法。

70 年代初, 人们对结构程序设计的基本原理及逐步求精方法进行了多方探讨及实际应用, 并进行了卓有成效的实践。H. D. Mills 提出了一般意义上的自顶向下开发方法。1970 年 W. Royce 提出了对软件工程具有重要意义的体现自顶向下开发风范的模型——**瀑布模型**, 提出了**软件生存周期**的概念, 把软件工作涉及的范围划分为需求分析、软件设计、编码测试及运行维护等几个阶段, 成为软件开发中之有效的模型。从此, 软件开发方法就不局限于编程阶段, 而对编程前的分析和设计以及后期的维护等给予了更多的重视。在结构程序设计的基础上提出了结构化分析和结构化设计, 形成了结构化开发方法。

E. Yourdon 和 T. DeMarco 等人提出了用数据流程图 (DFD) 表示系统逻辑结构的结构化分析 (SA) 方法和用结构图 (SC) 表示软件模块结构的结构化设计 (SD) 方法。这种结构化开发方法是以数据流分析和软件的功能分解为特征的。与此同时还产生了侧重于数据结构分析的面向数据结构的一类开发方法。典型的有 K. Orr 和 J. D. Warnier 的**数据结构化系统开发方法 (DSSD)** 和 M. A. Jackson 的**系统开发方法 (JSD)**, 以及 D. T. Ross 的**结构化分析设计技术 (SADT)** 等。

基本内容

自顶向下方法源于结构程序设计, 结构程序设计方法的核心就是逐步求精和自顶向下, 它的引入

从方法学上说是程序设计技术的一场革命,其意义不仅在于当时为解决“软件危机”的实用性一面,而更深远的意义还在于它使程序设计技术成为一种系统的方法,使程序设计从依赖设计者个人的技艺变成一门科学。

逐步求精方法和 Mills 的一般意义上的自顶向下开发方法的基本内容是这样的:首先研究和设计整个系统的结构以及各个子系统之间的关系,并编写和测试总控程序,然后再分析各子系统内部的功能,编写和测试各功能模块,各功能模块分步地联入整个系统中。

结构化分析(SA)方法 SA 方法是在需求分析阶段使用自顶向下、逐层分解的方法,即在顶层抽象地描述整个系统,然后逐层分解,到底层时再具体地描述系统的每个细节。该方法用一套分层的数据流程图表示系统的逻辑模型,用数据字典表示数据特征,用判定表、判定树、结构化语言表示处理的逻辑。分层数据流程图和分层定义数据的方式具体体现了自顶向下逐步求精的过程。

结构化设计(SD)方法 SD 是在结构化分析的基础上,用模块化的方法,自顶向下,根据系统的规约从逻辑模型得出具体的设计方案。该方法的主要工具是描述系统控制结构的模块结构图(SC),它反映软件各组成部分的相互调用关系和相互影响。

数据结构化系统开发(DSSD)方法 这种方法利用顺序、选择和重复三种构造成分来表示分层的信息进而导出软件的结构,使用的基本工具是 Warnier 图。它首先从信息的产生者和接受者的观点,观察数据如何在两者之间流动确定应用环境,然后表达应用功能和给出应用结果。构造 Warnier 图的过程及 Warnier 图的层次结构都体现了自顶向下方法的基本原则。

Jackson 系统开发(JSD)方法 JSD 是在其结构化程序设计(JSP)方法的基础上发展起来的。它以数据结构为基础,通过一组映射或转换来建立软件系统的结构。方法分为实体动作分析、实体结构分析、初始模型定义、功能描述、时间特性的确定和编程实现等几个步骤。

结构化分析与设计技术(SADT) 该方法要求自顶向地下建立 SADT 模型,而模型由一组有序的结构分析图构成,每张结构分析图是由若干个结点以及连结这些结点的弧组成的工程图纸。图有两类,一类称为活动图,一类称为数据图。活动图中的结点表示活动而数据图中的结点表示数据。

总结与展望

自顶向下方法是软件工程和软件方法学的重要内容,这类方法以及与其相应的支持工具已在各类软件的开发中发挥了重要的作用,这类方法基于“把整体划分为相对独立的规模较小的组成部分,比作为整体实现代价小”的软件工程原理,有其独有的优越性。但是这类方法也有它的局限性,自顶向下方法共同存在的缺陷是高层设计对全局影响过大,这就使得如果高层设计的不合理或软件需求发生变化,软件改动的成本太大,甚至要从头设计,带来维护方面的巨大困难。从而软件的重用技术以及提高软件易维护性就成为近代发展的各类自顶向下与自底向上相结合的方法以及面向对象的开发方法所追求的主要目标。

参考文献

- Mills H D. Top-down Programming in Large Systems. In: Rustin R (ed). Debugging Techniques in large Systems. New York: Prentice - Hall, 1971. 41 ~ 55
(郝克刚 鱼滨)

zidong biao yin

自动标引(automatic indexing) 使用计算机自动对文献赋予检索标识的过程和技术。标引工作是信息检索中最重要和最困难的工作之一。标引过程是一个智力活动。传统做法是由标引专家人工完成。但人工标引有一致性差、费用高和效率低的严重缺点。实现自动标引是实现现代信息检索系统的重要目标。

在全文检索系统中,文献中除了没有检索意义的高频词(通常称停用词或禁用词)以外的每个词都认作标引词,自动提取这些词的过程称为**全文自动标引**。高质量自动标引的目标是找出文献中最重要的概念,并赋予不同的权值。

自动标引的研究起源于 H. P. Luhn 在 20 世纪 50 年代末对文献中词的分布规律的研究。几十年来,在单词标引、短语标引和语义标引等方面都进行了深入的研究。单词标引的目标是找出文献中最主要的单词。单词标引的方法有 3 种:①基于词频统计的标引;②基于词区分值的标引;③基于概率加权的标引。基于词频统计的方法简单易行,其中逆文献词频对检索效果的改善已被公认。短语标引的目标是分析文献中“名词+名词”或“形容词+名词”结构的短语以克服单词标引专指度不够的缺点。一般有两种方法:一种是基于词相关出现频度

的统计学方法;另一种是基于句法分析的计算语言学方法。两种方法的识准率基本相同。语言学方法的识全率较高一些,但语言学方法对领域的适应性较差。语义标引的目的是借助于自然语言理解技术,获得对文献的深层理解。由于信息检索系统中领域的无关特性,语义标引有很大困难。主要的研究内容包括省略、指代的处理、布尔逻辑的语义解释、概念语言和词的语义分类等。一部带有词性、句法、语义和语用知识的词典是语义标引的基础工作。

和英文系统相比较,汉语自动标引还有自动分词的问题,需考虑信息检索用的分词规范和计算语言学中分词规范的差异,例如,在信息检索中,通常把一些词组作为单个检索词,而在计算语言学中,这类词组必须切分为多个词。迄今为止,建立了约十几个汉语自动标引系统,个别的投入了实用,其基本技术大都基于自动抽词标引,离智能标引还有很大距离。较为先进的自动标引一般包括如下过程:分词,停用词过滤技术(汉语中的停用词处理较西文复杂),低频词归类处理,高频词组词处理,最优加权函数设计,基于句法分析或词相关频度的短语生成,语言学分析(如指代、省略等),蕴涵概念的推理等。一般认为基于统计学方法和计算语言学方法的结合是实现较高水平自动标引的现实途径。

自动标引更进一步的工作是对文献进行自动分类,有效的文献分类可以改善检索的效果和提高检索效率。目前自动分类以统计学方法为主,辅以语言学知识。自动分类对处理的领域有限制,在比较规范或比较窄的专业领域,可以取得较好效果。

参考文献

1. Salton G, McGill M J. Introduction to Modern Information Retrieval. McGraw Hill, 1983
2. Salton G. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison Wesley, 1989
3. Frakes W B, Yates R B. Information Retrieval: Data Structures and Algorithms. Prentice Hall, 1992
(施水才)

zidong hedaiqu

自动盒带库 (automated cartridge tape library) 一种以盒式磁带为基本存储单元,能对大量盒式磁带进行自动管理的联机大容量存储系统。它是利用盒带驱动器具有自动装卸盒带功能的特点加上能灵活存取带盒的机械手与自动管理软件

实现盒带库的自动管理的。其基本构成部分主要有存放大量盒带的仓库(盒带架)、盒带驱动器与控制器(读写站)、机械手与带盒存取控制器、系统管理处理器与管理软件以及与主机的接口等。

盒带仓库有圆形与长形两种常见的形式。在圆形盒带仓库中,盒带架沿圆周内外两层排列(外层的内侧与内层的外侧),机械手在两层盒带架的中间绕圆心作 180° 左右旋转。同时,机械手可上下升降,以便到达所需存取的盒带位置。然后伸向盒带,将它从架上取出。机械手携带盒带转至驱动器的位置,将它放入驱动器的装卸窗口。驱动器自动完成带的加载,然后便可随时进行联机读写。在长形盒带仓库中,盒带架沿长的方向分前后两侧排列,机械手沿直线轨道移动。其余动作则与上述的类似。盒上印有条形码标志。取放带盒的手臂上装有光源和光学装置,用来识别盒上的条形码。

1986年以来,陆续出现了许多种类型的自动盒带库。其中以IBM 3480型半英寸氧化铬盒带技术为基础发展起来的自动盒带库较成熟,应用较普遍。其他类型的大容量盒带,如家庭录象用半英寸VHS型盒带、8 mm视频盒带、19 mm D1/D2型盒带都被用来作成自动盒带库。美国STK公司1988年首先推出了利用3480型盒带的自动盒带库系统,一个库最多可收容6 000盒带,总存储容量为1.2 TB。多个库可连接在一个系统中。IBM公司1992年推出的3495型带库数据服务器使用了3490型盒带驱动器,库的收容量为5 660~18 920盒,总的存储容量为2.3~7.6 TB。日本Sony公司的DMS-700M系统收容了19 mm D1-M型盒带736盒,总容量为30 TB。

一种应用8 mm盒带的小型自动盒带库,采用圆形转台的结构,盒带的存取是靠转台的转动来实现的。一个小型转台可收容50多盒8 mm的盒带,8个转台叠加在一起可连成一较大的系统,构成2 TB的存储容量。

盒带的平均存取时间(即机械手平均取放盒带的时间)大约在10 s的数量级。

参考文献

1. Wood T, D -I Through DAT. 9th IEEE Mass Storage Symposium, 1988; 130~138
2. 板生清等,磁気テープ記憶自動化システム的方式構成. 通信與計算機研究実用化報告, 1986, 35(7):41~48
(刘锡刚)

zidongji lilun

自动机理论 (automaton theory) 以离散数学系统的结构、功能以及两者之间关系为主要研究内容的理论。

自动机可分为以下 11 种,各有其特点与作用。

有限自动机 一类满足下列条件的离散动态系统:①在考虑的每一时刻系统按输入隶属于有限个可能的状态。②输入从可能状态中选择有限个状态。③任一时刻系统的状态由输入状态和前一时刻系统的状态所确定。电话交换机、升降机、数字电路、神经网络、计算机这类存储量有限的机器都是有限自动机的实例。主要研究(包括开关网络)结构、功能的综合与分析,即讨论功能的数学描述语言(相对应的正规语言)、文法及其性质,状态化简与标准化,状态赋值,各种有限自动机(确定的、不确定的,单向的、双向的,带输出、不带输出的,线性的,自治的,可逆的)的等价性,并设计出满足要求的逻辑网络,有限转换器等。有限自动机已建立了比较完善和系统的研究方法(包括概率、代数、模糊数学方法),是自动机理论的基础。应用于自动控制,生物系统,语言学,通信理论,编码理论,神经网络,计算机高级程序语言的编译设计、词法分析、文本编辑,操作系统,模式识别等。

下推自动机 单向非确定的识别程序,由有限个状态控制一条输入带和一个先进后出的下推表(栈)所组成。有限自动机类是下推自动机类的真子集。主要研究下推自动机的变形和子类(如确定的、双向的、加速的、栈的元素个数的变化等,特别重要的是确定的下推自动机),相对应的上下文无关文法、语言及其运算,不可判定性,下推转换器等。与**有限自动机**一起应用于句法制导翻译模式,比有限自动机更适合于描述各种计算机高级语言。

线性有界自动机 合于条件:①包含输入带左右两端标志的输入字母表。②不能在左(右)端标志向左(右)移动,也不能在左右两端标志处打印任何符号的不确定图灵机。主要研究相应的上下文有关文法、语言及其性质,不可判定问题等。线性有界自动机类真包含下推自动机类且是图灵机类的真子集,还存在着许多公开问题尚待解决。

图灵机 描述通用计算机计算能力的数学模型。主要研究各种构造技术与变形(如无穷输入带的单向、双向扫描、多带、多维、多头、多栈、离线、带外部信息源、状态和符号数受限等)的等价性问题,与**波斯特机**的等价问题,对应的递归可枚举语言性

质和文法,停机问题,不可判定问题,丘奇-图灵论题:所有“可计算”函数恰好是部分递归函数,即图灵机可计算函数,通用图灵机等。图灵机可以作为枚举器,是整数函数计算机,在**可计算性理论**和**计算复杂性理论**中有着广泛的应用。图灵机类以线性有界自动机类为其真子集。证实了存在非递归可枚举语言。

时序机 输出与转移函数、转移状态有关的**有限自动机**。主要研究状态的完全化,极小化,时序转移函数的归约与分解,线性时序机,概率时序机,单式前缀、幺半群对正规语言的分解算法与复杂度。应用于研究布尔矩阵的时序开关电路网络的综合与分析,确定冗余性与可靠性,设计计数器等。

波斯特机 由一些基本语句组成的程序框图。它只有一个变量,该变量的取值是字母表 Σ 中字母组成的串(即 Σ^* 的元素)。框图中的语句包括:①开始语句(只有一个)。②停机语句(接受语句,拒绝语句两个)。③分支判断语句。④赋值语句。 Σ^* 的任何元素都可以作为波斯特机的输入,若运行结果在接受(拒绝)语句处停机,表示波斯特机接受(拒绝)该串。 Σ 中有一个特殊字母用 # 表示,它在波斯特对应问题(即同一字母表上任意两个相同长度的字有无匹配的问题)的变形和证明波斯特机与图灵机有同样的计算能力起着重要的作用。波斯特对应问题广泛应用于各种不可判定问题,例如任一上下文无关文法是不是多义这个问题是不可判定的。

随机存取机 由无限多个内装任一整数的存储器 R_0, R_1, \dots 和有限个内装任一整数的算术寄存器组成,整数均译码为计算机指令。选择适当的指令集编程随机存取机可以模拟现有的任一计算机。提供了基本指令随机存取机与图灵机可相互模拟,是算法分析和**计算复杂性理论**中重要的串行计算模型。主要研究变址形式,从理论上分析计算机串行程序的时、空资源耗费,虚拟并行时间(巡回)实现的变换,判断寄存器、变址器内容是否为 0 实现条件转移等。

栈自动机 满足条件:①输入头为双向,但只读到结束记号。②只读模式时,栈顶压入或弹出动作的同时,栈头可在栈内上下(或左右)扫描但不重写任何符号的**下推自动机**。主要研究各种栈自动机(如确定的,不确定的,可擦符号的,不擦符号的——即永不弹出栈符号,单向的——输入头永不左移,对输出字母表进行考虑,以检验只读可擦输入

寄存器是否为空,且栈装有指针,指针可以双向移动,当指针在最下(右)位置时,允许擦去,并打印栈下(右)边的符号,双向的等等),转移表,它所接收、识别的索引语言及其时、空复杂类的特征。应用于推广的上下文无关语言和文章结构。

无限自动机 有限自动机作控制器和存储不受限制,某种数据结构刻画的外部环境组成,是现实计算机的理想化模型。主要研究在计算时、空或其他资源的限制下,自动机所计算的函数类和识别的语言类的描述、包含关系和代数性质。模拟(通过简单的编码,在一个自动机上实现另一个自动机的功能,如图灵机和其他许多自动机可互相模拟)、通用性、不可判定性等。这些都有助于理解计算过程的本质,以及估计算法设计的好坏和固有复杂性下界。

概率自动机 对自动机所处的状态在输入某一字母时下一动作规定条件概率,并给出初始状态的概率分布。动作包括状态的转移,改写字母或输出。主要研究环境或内部具有无限或有限个随机因素的离散数字系统,讨论的参数为可数个,要求电话服务的过程就是这种系统的实例。研究方法一方面推广已有的自动机结果,平行地得到概率图灵机,概率时序机,随机语言类的代数性质(如对某些运算的封闭性),另一方面也考虑了不少新问题,如用 z -变换确定自动机的响应矩阵;估计斯图哈斯蒂矩阵的 k 阶转移应用于多步决策过程;引入自动机的熵和特征值,概率结构的树自动机,应用于模式识别、可靠性问题、动态规划等。

细胞自动机 大量互连有限自动机的集合;部件或小计算机互连成大计算机、并行工作部件或计算机的理论模型。将每个自动机想象为一个细胞,由它们构造的子孙自动机至少要有其双亲的功能。20世纪50年代初,冯·诺依曼研究自繁殖复杂机制自动机的逻辑问题时,设计了细胞空间,其中每个细胞都具有相同的领域,不随时间而改变,细胞间的物理距离不影响结果的判断,赋给细胞共振机制并由中央传感器控制,形成庞大、高度分布、同步操作的逻辑成分。根据他提出的概念,发展出许多研究方向,主要有:①适合空间,②棋盘格空间,③图细胞自动机,④动态细胞自动机—— L 系统,⑤Holland迭代电流计算机(其各细胞有大量状态,邻域可以不同,且随时间可改变),⑥通用计算机构造器(细胞空间及其配置依靠在线交互作用简化)。研究与形式语言的关系,具有一致结构的细胞自动机的综合、分析和容错也是研究的重要内容。应用于超大

规模集成电路、并行识别器和并行计算机的设计,模式识别,生物发育,生物化学(开发宏分子和微有机组织,提供研究条件和性质)等。

参考文献

1. 中国大百科全书·电子学与计算机卷. 北京:中国大百科全书出版社,1980
2. Hopcroft J E, Ullman J D. Introduction to Automata Theory, Languages, and Computation. Reading, Mass: Addison-Wesley, 1979
3. Arbib M A. Theories of Abstract Automata. Englewood Cliffs, N. J.: Prentice-Hall, 1969

(张一立)

zidong tuili

自动推理 (automated reasoning) 在计算机支持下实现的推理。它是人工智能领域中的重要研究课题。在20世纪60年代中期以前,定理机器证明的注意力还仅仅限于数学方面。从60年代后期开始,定理机器证明领域的研究者已开始将注意力转向数学以外的其他领域,诸如程序自动生成、逻辑程序设计以及更一般的智能系统中的推理问题。

定理机器证明的研究是自动推理领域中的先驱性工作。70年代专家系统和知识工程的出现,使人们认识到,仅仅研究从真前提得出真结果的古典推理方法是不够的。因为人类面对的是一个充满不确定信息的环境。人类在这种环境里进行着有效的思考和推理。因此,为了建立类似于人的智能系统,研究那些更接近人类思维方式的推理,例如,非单调推理,模糊推理等等,就变得越来越必要了。

目前自动推理的研究,一方面,表现在专家系统中,各种面向特殊问题的推理方式的研究,例如, DENDRAL系统的用于化学合成的推理, PROSPECTOR的用于地质方面的推理, MYCIN的用于医疗诊断的推理等等;另一方面,在计算机辅助推理的研究上,也取得成果。例如,以L. Wos为首的小组在美国Argonne国家实验室建立的AURA系统,这个系统已经帮助人们回答了以前在数学和形式逻辑方面的一些未解问题。随之而来的,面向自动推理的程序语言,如PROLOG,也引起了研究者的兴趣。

自动推理领域的研究还是初步的,诸如领域知识的表示,推理规则的控制策略等重要问题,都没有满意的解决。一个明显的例子是:人能进行迅速推理的很多问题,用自动推理程序时,仍要花费大量时间。推理程序和人之间的通信仍很困难,自然语言

处理方面的工作和自动推理的进展有重要关系。L. Wos 认为解决自动推理的关键是利用现在和将来的推理程序对已经解决和尚未解决的问题进行实验,从而导致新的推理规则和有效策略的发现。例如,对数论问题求解的尝试,导致 L. Wos 发现了调解方法。

自动推理包含下面一些研究内容:定理机器证明、程序正确性验证、程序自动生成、逻辑程序设计、非单调推理、模糊推理、约束推理、定性推理、类比推理、归纳推理、自然演绎法、归结方法、重写方法、吴方法等等。

自动推理的近期目标是:得到各种推理程序,它们中的每一个都相当于一个自动推理助手,人们应该能有效地和这个助手“交谈”。远期目标应该是:当你向这样一个程序提出问题后,你就可以去考虑别的问题了;当你再回来时,原来的问题已经解决了,丝毫没有你的帮助和干涉。

日本在执行其五代机研制的十年计划中得到如下一个看法:新一代计算机应该是一种新概念下的计算机。它的机器语言应该是像 PROLOG 语言那样的谓词逻辑符号串。用逻辑表示信息,用逻辑推理解决问题将是新一代计算机的特征。因此,这种计算机的基本操作不再是算术运算,而是逻辑推理。所以,自动推理的研究,对于未来一代新型计算机的设计是很有必要的。

参考文献

刘叙华. 基于归结方法的自动推理. 北京: 科学出版社, 1994

(刘叙华)

zijiyan dianlu

自检验电路 (self-checking circuit) 不需外加测试码而能自动地证实电路本身或系统中是否存在故障的电路。

假设电路有 p 个输入端和 q 个输出端, 则可能出现的输入组合为 2^p 个, 称为输入空间, 可能出现的输出组合为 2^q 个, 称为输出空间。如果在正常运行时输出空间中的 2^q 个组合均可能出现, 则就无法通过观察和验证输出值来确定电路是否存在故障。如果在输出空间 2^q 中仅有 $k < 2^q$ 个输出组合为有效的, 那么在出现另一些属于 $2^q - k$ 中的组合时就可断定电路中必定存在故障。通常称属于 k 中的组合为有效的输出组合, 称为合法码字, 否则, 即为错误的输出组合, 称为非法码字。

若电路中任一个故障 f 出现时, 都能使该电路

的输出成为非法码字, 则称该电路为故障安全电路。

若对电路中的任一个故障至少存在 1 个输入组合使得该故障发生时其相应的输出组合为非法码字, 则称该电路是自测试的。

若一个电路既是故障安全, 又是自测试的, 则称该电路为完全自检验电路。

在校验电路中, 通常设置 1 个单输出的校验器。当电路正常时, 校验器输出为 0; 在发生故障时, 校验器输出为 1。这样就起到了校验电路的作用。但是这时也必须注意, 校验器本身的输出端是否存在固定 1 的故障。因为当校验器输出发生了固定 1 的故障后, 就会对被校验电路产生错检或漏检现象。

自校验电路通常被用在如铁道系统等信号灯、安全灯控制系统中以防止信号的误发等。

参考文献

1. Breuer M A, Friedman A D. Diagnosis & Reliable Design of Digital Systems. Computer Science Press, Inc., 1976

2. Fujiwara H. Logic Testing & Design for Testability. The MIT Press, 1985

(徐拾义)

ziran jingwu zaoxing

自然景物造型 (modeling of natural phenomena) 在计算机系统中生成自然界景物模型的过程、方法和技术。许多复杂的自然景物, 如山峦、树木、草丛、烟雾、火等, 由于其形状的不规则性, 很难用欧氏空间的连续函数来描述, 但可以通过一些能近似反映其形状规律的计算过程来模拟。这类造型技术的主要特点是能够逼真地再现自然景象, 而不要求模型与实际景物完全一致。常用的方法有分数维造型技术、基于文法的造型技术、粒子系统等。

分数维造型技术是用递归方法来模拟具有自相似性(从严格意义上或从统计意义上)的、随机的复杂形体的一种造型技术。用这种技术可以逼真地描述复杂静态景物, 如海岸线、山峦等。这种方法最早出现于 1904 年, 19 世纪 60 年代 B. B. Mandelbrot 提出了分数维的概念。

基于文法的造型技术采用正规文法生成结构性较强的形体拓扑结构(如树木), 再通过几何解释来形成形体的真实感图形(如树干、树叶)。该方法是 1984 年 A. R. Smith 为模拟植物而引入的。

粒子系统是采用大量随时间变化的粒子图元来

描述自然景物的一种造型方法。粒子系统中的每个粒子随着它寿命的增加,可以产生新的粒子,亦可以得到新的属性,也可以消亡。粒子还可以遵循确定的或随机的运动规律改变其位置。最早的粒子系统是由 W. T. Teeves 于 1983 年提出来的。

参考文献

1. Foley J D, Dam A V, et al. Computer Graphics: Principles and Practice. Second Edition. Addison-Wesley, 1996

2. 孙家广, 杨长贵. 计算机图形学(新版). 北京: 清华大学出版社, 1995 (杨长贵 金小刚)

ziran yanyi fa

自然演绎法 (natural deduction method)

一种模拟人的演绎思维过程,在计算机上实现定理证明的方法。W. W. Bledsoe 于 1975 年提出。这种证明方法与定理机器证明的传统方法——归结方法在很多方面有本质的区别。

首先,自然演绎法证明的主导思想与归结方法不同。假设我们要证明的公式是 $H \Rightarrow C$, 其中 H 是前提集合, C 是结论集合,归结方法实际上采用的是反证证明方法,转去证明 $H \wedge (\sim C)$ 的不可满足性。而自然演绎法却是直接证明 $H \Rightarrow C$ 的恒真性,其演绎过程与人们证明这个问题的思维过程十分类似,自然演绎这一名称也即由此而来。

因为证明的主导思想不同,自然演绎法在对一阶逻辑公式的 Skolem 化的过程上也与归结方法不同。归结方法的 Skolem 化的结果是全称量化的,存在量词用 Skolem 函数代替,而自然演绎法恰恰相反,Skolem 化的结果是存在量化的,全称量词用 Skolem 函数代替。

归结方法从反证的角度出发,对前提条件和要证明的结论不加区别。而自然演绎法却在证明中始终保持它们的区别,不断地分析前提和结论的逻辑结构,根据它们的结构选用不同的演绎规则。例如,假设要证明的公式具有 $H \rightarrow (P \wedge Q)$ 的形式,则自然演绎法根据结论的合取形式,把要证的公式分裂成两个子目标 $H \rightarrow P$ 和 $H \rightarrow Q$,证明了两个子目标后再综合出整个命题的证明。在证明过程中,归结方法只使用一条规则——归结规则,而自然演绎法却可以有好多条演绎规则。

自然演绎法的优点是演绎过程中所产生的子目标的数量较少,因而占用的存储空间也较少,证明问题的速度也比较快。中等难度的问题在人们容许的

时间内即可得到证明。由于它的演绎过程和对问题的证明过程类似,因而便于人机对话,实现交互式的证明。例如,如果机器证明的时间过长或者在证明过程中遇见问题时,可以中断证明,请求人的帮助与指导。

自然演绎法的弱点是它的不完备性——有些人利用通常的推理方法能够证出来的定理,使用自然演绎法却证不出来。为了解决自然演绎法的不完备性问题,人们对自然演绎规则提出了各种修改意见。这些修改虽然能够解决自然演绎法在某些具体问题上的不完备性,却不能从根本上彻底解决它的不完备性。自然演绎法的不完备性之有效解决,至今仍是它的一个大问题。

参考文献

Bledsoe W W, Tyson M. The UT Interactive Prover. University of Texas, 1975 (姜云飞)

ziran yuyan chuli

自然语言处理 (natural language processing, NLP)

研究计算机通过人类熟悉的本族语言同用户进行听、说、读、写等交流的技术。早期又叫自然语言理解(NLU),更强调对人类自身语言能力和思维本质的探索,希望这有助于造出能够模仿人类高级智能行为的机器。因此,从一开始它便是人工智能学科的一个重要分支,是语言学、计算机科学、认知科学、信息论和数学等多学科基础上形成的一门新兴的边缘学科。

正如什么叫“智能”一样,对于“理解”这一术语也存在着各式各样的认识。然而在人工智能界,人们普遍认为可以采用如下的“图灵试验”来判断计算机是否“理解”了自然语言。

(1) 问答 机器能正确摘取输入文本所承载的信息,并据此回答有关的问题;

(2) 释义 机器能用不同的词语和句型来复述输入文本的内容;

(3) 文摘生成 机器有能力生成输入文本的摘要;

(4) 翻译 机器具有把一种源语言的输入文本翻译成另一种指定目标语言的能力。

其实,这些任务都可以发展成为有商业价值的自然语言应用系统。

40 年来自然语言处理的研究大体上经历了 3 个时期,即 20 世纪 60 年代以关键词匹配为主流技术的早期,70 至 80 年代以句法-语义分析方法为主

流的中期和90年代后以语料库和统计语言模型为主流技术的近期。早期的自然语言理解系统大多没有真正的语法分析,而主要依赖关键词匹配技术来识别输入句子的含义。设计者在这些系统中事先存放了一批包含某些关键词的模板,每个模板都与一个或多个解释(即响应式)相对应。系统将当前的输入句子同机储的模板逐一匹配,一旦匹配成功便获得了这个句子的解释,而不再理会句中那些非关键词语对句义的影响。这种分析技术的不精确性往往导致错误的理解。这一时期的著名系统有SIR, STUDENT和ELIZA等。

进入70年代以后,一批采用句法-语义分析方法的自然语言处理系统脱颖而出,它们在语言分析的难度和深度方面都比早期系统有了长足进步。这些进步很大程度上得益于计算语言学在语言理论方面的成果。如1972年W. Woods设计的LUNAR系统采用了他本人首创的扩充转移网络(ATN)语法;同年,T. Winograd设计的“积木世界”(SHRDLU)系统,采用的是M. A. K. Halliday的系统功能语法;而Roger Schank研制MARGIE系统时,则以他本人建立的概念从属理论为句义表达和推理的基础。这些语言理论建立在词类和短语类符号的基础上,具有较高的抽象能力,但对句子结构的描写过于粗糙,不能有效地覆盖真实文本的语法现象。80年代起,计算语言学家努力对语法理论加以扩充和改造,相继提出了词和短语描写的复杂特征集和合一语法,以及词汇主义等重要理论。这个时期的著名语法理论有广义短语结构语法(GPSG)、词例功能语法(LFG)和中心词驱动的短语结构语法(HPSG)等。这些新的语法理论大大加强了语法的约束力,把语法对语言规律的描写重点从普适的句法规则转移到具体词例的个性规则上来,俗称“大词库,小语法”。尽管如此,用这些语言理论实现的自然语言系统只在少数受限领域中得到应用。例如1976年加拿大蒙特利尔大学与加拿大联邦政府翻译局联合开发的实用化机器翻译系统(TAUM METEO),提供了天气预报的英-法翻译服务。但这个翻译系统的词典中只有约3000个词条,其中超过半数地名和气象专业术语。当需要面向其他领域或扩大规模时,这类系统暴露出许多弱点,如开发周期长、语言覆盖面窄和系统鲁棒性差等,因而远不能满足社会信息化对大规模真实文本处理的迫切需求。

90年代以来,大规模语料库和基于语料库的统计语言模型技术首先在言语识别、信息检索(搜索

引擎)等实用化系统中获得成功,并迅速推广到浅层句法分析、信息抽取、自动文摘、文本挖掘和机器翻译等自然语言处理系统的研究和开发中。这种方法的最大特点是把一个自然语言处理的任务看成是一个机器学习问题,而不是一个传统的句法-语义分析问题。换句话说讲,解决任何这样一个任务的数学模型的众多参数可以通过带有特定标注的训练语料来估计。这就缓解了句法-语义方法所遭遇的困境:一是知识粒度精细化,大大提高了语言知识的覆盖面;二是实现了语言知识获取的自动化或半自动化,突破了知识获取的瓶颈问题。显然,这个时期的主流技术更注重语言处理,而不是语言理解。因此,90年代起“自然语言理解”这个术语已日益被“自然语言处理”或“语言工程”所取代。这标志着计算语言学界在理论、方法和工具等方面都正酝酿着一场巨大的变革。一方面,技术上的这些重大趋势应当引起研究人员的足够重视;另一方面,也不可否定句法-语义方法在自然语言分析和生成方面的贡献。人们期待着多种方法的繁荣和融合必将促使自然语言处理的研究与开发走向更大的成功。

参考文献

1. Christopher D. Manning and Hinrich Schütze. Foundation of Statistical Language Processing. Cambridge, Massachusetts: The MIT Press, 1999
2. 黄昌宁,李涓子. 语料库语言学. 北京:商务印书馆,2002
3. 石纯一,黄昌宁,王家欣. 人工智能原理. 北京:清华大学出版社,1993 (黄昌宁)

ziran yuyan chuli de cifa fenxi

自然语言处理的词法分析 (morphological analysis of natural language processing)

通过语素这一结构成分对词的结构、形式和类别所做的分析。又称形态分析。在传统上,它跟句法学相区别,后者研究句子中词与词之间的组合规则。

自然语言理解系统中,在对输入句子进行句法分析之前,一般都需要对句子中的词进行词法分析,以便根据词的原形(如英语中的动词不定式、名词的单数形式等)去查询电子词典中相应词条下登录的词法、句法和语义信息。在汉语、日语等以字符连写的语言中,在词法分析阶段必须完成词语的切分,才能按词去查询电子词典。

由于汉语的词和短语之间几乎没有形态变化可以作为辨识的依据,加之长期以来汉语学界对词的

界定缺乏统一的认识,使我国汉语自动分词的研究步履维艰。尽管如此,近十年中我国仍在现代汉语自动分词的研究中取得了一批接近实用化的成果,如北京航空航天大学等单位主持制定的《信息处理用现代汉语分词规范》已作为国家技术标准颁布。一般来说,自动分词算法可粗分为以下两大类:

(1) 机械匹配法 分词系统把输入文本中指定长度的字串与词表中的词逐一匹配,若匹配成功便确认该字串为一个词。这类方法包括正向最大匹配法(MM)和逆向最大匹配法(RMM)、双向扫描法和基于统计的方法等。

(2) 基于理解的分词方法 整个系统由词表、知识库和推理机三部分组成。知识库中存放着各种语法规则和分词知识,推理机综合运用词表和知识库提供的词类、句法和语义等语言知识来实现自动分词。这类方法包括有穷多层次列举法、邻接约束法、联想回溯法和分词专家系统等。

在自动分词系统中引起切分错误的主要原因是:

(1) 歧义切分字段 又分为交集型和多义组合型两种。如果 $\alpha\beta\gamma$ 是一个歧义切分字段,其中字串 β 既可以与字串 α 构成词 $\alpha\beta$,又可以与字串 γ 构成词 $\beta\gamma$,就称它为交集型歧义切分字段。例如,“美国会”可以有“美/国会”或“美国/会”两种可能的切分,因此是交集型的歧义切分字段。又如 $\alpha\beta$ 是一个歧义切分字段,其中字串 α 和 β 都是词,而且 $\alpha\beta$ 也是词,就称之为多义组合型的歧义切分字段。其实例如“才能”、“把手”、“马上”、“将来”等。显然,歧义字段的正确切分依赖于所在句子的上下文(即语境)。

(2) 未登录词 如人名、地名、机构名、译名和新词等未曾登录在词表上的词。未登录词所引起的切分错误也是相当可观的。

当前汉语自动分词的研究主要集中于上述这两个难点上,这是汉语自动分词系统真正走向实用化的关键。

参考文献

1. 石纯一,黄昌宁,王家庶. 人工智能原理. 北京:清华大学出版社,1993
2. 刘开瑛,郭炳炎. 自然语言处理. 北京:科学出版社,1991 (黄昌宁)

ziran yuyan chuli de jufa fenxi

自然语言处理的句法分析 (syntactic analy-

sis of natural language processing) 根据系统已知的语言知识确定输入句子的句法结构,即识别构成句子的各个成分以及它们之间的相互关系的过程。例如,确定句子中述语动词的主语和宾语,中心词及其修饰成分等。句子的句法结构通常以句法树来表示。

句法分析总是和一定的语法模型相联系。印欧语言通常采用短语结构语法(即所谓层次分析法)来描述一个句子的句法结构。例如,句子:

他 给 我 三
 〈*代〉 〈*动〉 〈*代〉 〈*数〉
 本 书
 〈*量〉 〈*名〉

的句法树可以用短语结构语法表示如图1。图中,

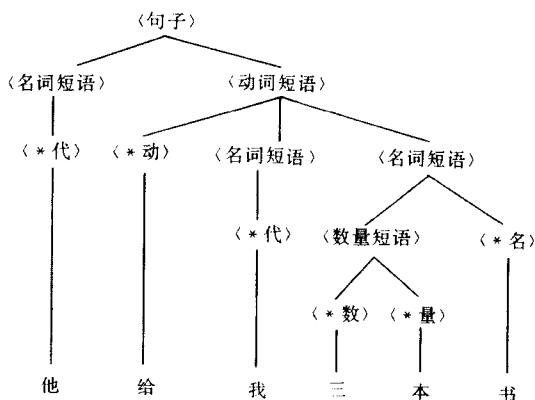


图1 用短语结构语法表示的句法树

〈×〉表示短语标记,也是语法的非终结符;〈*×〉表示词类,是语法的终结符;最下层是输入句子的词。该语法的短语结构规则如下:

〈句子〉→〈名词短语〉 〈动词短语〉
 〈名词短语〉→〈*代〉
 〈名词短语〉→〈数量短语〉 〈*名〉
 〈动词短语〉→〈*动〉 〈名词短语〉
 〈数量短语〉→〈*数〉 〈*量〉

由于汉语中短语类型与句法功能不存在一一对应的关系,所以在传统语法中,常用主语、谓语、述语、宾语、补语、定语、状语等句子成分来表达句子的结构,这就是句子成分分析法(又称中心词分析法)。于是上述例句的句法树又可相应表述如图2。

用句子成分分析法写出的句法规则是:

〈句子〉→〈主语〉 〈谓语〉

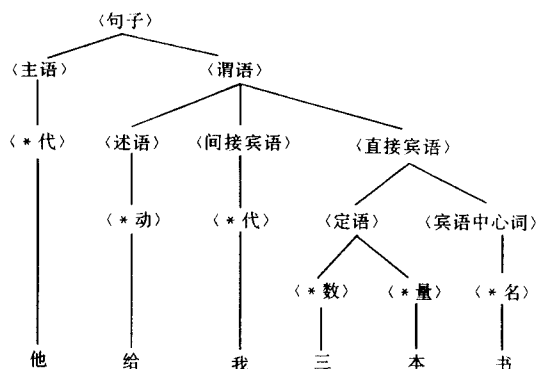


图2 用句子成分分析法得到的句法树

〈主语〉→〈* 代〉
 〈谓语〉→〈述语〉〈间接宾语〉〈直接宾语〉
 〈述语〉→〈* 动〉
 〈间接宾语〉→〈* 代〉
 〈直接宾语〉→〈定语〉〈宾语中心词〉
 〈定语〉→〈* 数〉〈* 量〉
 〈宾语中心词〉→〈* 名〉

上述两种语法模型都是规定性的,不符合语法规则的输入句子将被系统所拒识。而依存语法是描写性的,它认为一个句子中的任何两个词之间只存在一种依存关系,即必有一个是主词,另一个是从词,而且每个词只能从属于一个主词。依存语法还认为句子的述语动词是整个句子的主词,这种以述语动词为句法结构中心的语言观与现代语言学中有关动词的论元结构和格关系的学说是是一致的,因此受到学术界的普遍关注。上述例句的依存关系句法树可以用图3表示,图中子树“三本书”是“给”的直接宾语,而子树“三本”是“书”的定语。

比较上述的三棵句法树,不难发现依存关系句法树的层次最少,因为在本质上它直接描写了词与词之间的关系,如主语、宾语、定语等,没有短语标记

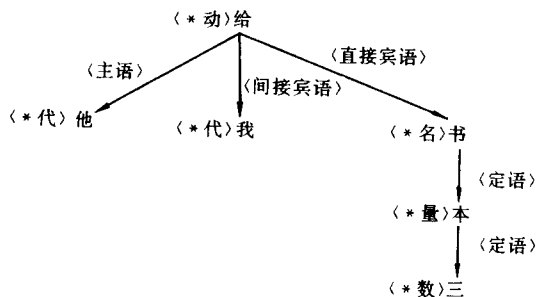


图3 用依存语法表示的句法树

一类的非终结符。

参考文献

1. 石纯一,黄昌宁,王家殿. 人工智能原理. 北京:清华大学出版社,1993
2. 刘开瑛,郭炳炎. 自然语言处理. 北京:科学出版社,1991 (黄昌宁)

ziran yuyan lijie

自然语言理解 (natural language understanding, NLU) 研究人类如何使用自身熟悉的本族语言与计算机进行信息交流,并探索人类自身的语言能力和思维活动的本质的活动。它是人工智能学科的一个重要分支。

自然语言理解的研究内容大体上与**自然语言处理**相当,都可以归结为对自然语言的句子和篇章(话语)这两个层次上的分析和生成的研究,但前者更着重于对“理解”的探索。正如什么叫“智能”一样,对于“理解”这一术语也存在着各式各样的认识。然而在人工智能界,人们普遍认为可以采用图灵试验来判断计算机是否“理解”了自然语言,具体的判断分述如下:

- (1) 问答 机器能正确摘取输入文本中的主要信息,并据此回答有关的问题;
 - (2) 释义 机器能用不同的词语和句型来复述输入文本;
 - (3) 文摘生成 机器有能力产生输入文本的摘要;
 - (4) 翻译 机器具有把一种源语言的输入文本翻译成另一种指定的目标语言的能力。
- 达到上述要求的计算机系统将在如下领域中获得广泛应用:

- (1) 机助人译 (MAHT) 或人助机译 (HAMT) 系统;
- (2) 自然语言人机接口或人机对话;
- (3) 信息理解 从用户需求的观点出发来摘取输入文本中的主要信息,进而把摘取的信息转换成机器内部的某种数据库存储格式并据此回答用户对该数据库的查询;
- (4) 自动文摘 未来的电子报刊将按照每个用户预定的内容和篇幅来编辑,然后按指定时间通过电话线直接送到客户的计算机终端屏幕上。因此,这种能根据需要生成长短不同的文摘的计算机系统将在电子报刊的发行中扮演重要角色。

三十多年来自然语言理解的研究大体上经历了

三个时期,即 20 世纪 60 年代以关键词匹配技术为主流的早期,70 年代以句法-语义分析方法为主流的中期,80 年代开始走向实用化和工程开发的近期。早期的自然语言理解系统大多没有真正的语法分析,而主要依赖关键词匹配技术来识别输入句子的意义。设计者在这些系统中事先存放了一批包含某些关键词的模式,每个模式都与一个或多个解释(即响应式)相对应。系统将当前的输入句子同机储的模式逐一进行匹配,一旦匹配成功便算得到了这个句子的解释,而不再理会句中那些非关键词语对句义的影响。这种分析技术的不精确性往往导致错误的理解。这一时期的著名系统有 SIR, STUDENT 和 ELIZA 等。进入 70 年代以后,一批采用句法-语义分析方法的自然语言理解系统脱颖而出,它们在语言分析的难度和深度方面都比早期的系统有了长足的进步。这些进步很大程度上得益于计算语言学在语言理论方面所取得的研究成果。如 1972 年 W. Woods 设计的 LUNAR 系统采用了他本人首创的扩充转移网络(ATN)语法;同年, T. Winograd 设计的“积木世界”(SHRDLU)系统,采用的是 M. A. K. Halliday 的系统功能语法;而 Roger Schank 研制 MARGIE 系统时,则以他本人建立的概念从属理论为句意表达和推理的基础。这些语言理论的应用一直延续至今。第三个时期的自然语言理解系统的最大特点就是实用化和工程化,其标志就是一批商品化的机器翻译系统和自然语言人机接口开始闯入国际市场。但延续了二十多年的基于句法-语义规则的自然语言理解系统也暴露出一些大的弱点,如开发周期长、语言覆盖面窄和系统鲁棒性差等,因而远不能满足信息社会对大规模真实文本处理的迫切需求。为了摆脱目前的困境,计算语言学界在理论、方法和工具等方面都正酝酿着一场巨大的变革,而语料库语言学和词汇主义的崛起则是这场变革的前兆。

参考文献

1. 石纯一,黄昌宁,王家底. 人工智能原理. 北京:清华大学出版社,1993
2. 刘开瑛,郭炳炎. 自然语言处理. 北京:科学出版社,1991 (黄昌宁)

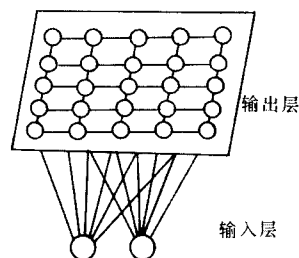
zizuzhi yingshe moxing

自组织映射模型 (self-organizing mapping model) 由输入层和输出层组成的两层式神经网络。通过设定的一种侧向抑制的简单竞争算法,实

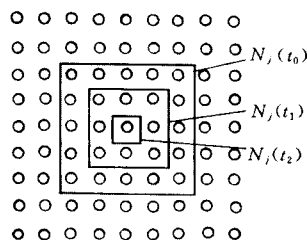
现类似于感觉信号映射到大脑皮层的有序特征映射,从而实现对输入样本的排序和分类。

神经生理学家很早就发现,厚约 2 mm,总表面积约 1.5 m^2 的大脑皮层是按功能分区的。例如有运动皮层、体感皮层和初级视觉皮层等。在听觉皮层区域内,神经元的排列反映出频率的对数刻度,低频信号引起该皮层一端区域的神经元反应,而高频信号则引起另一端的神经元反应。这表明,感觉信号的各种特征,是按照其取值的大小,有组织地排列在大脑皮层的一定空间范围内的。

80 年代初期由芬兰赫尔辛基大学信息科学系 Teuvo Kohonen 教授提出的自组织映射方法,就是用人工神经网络方法来实现这种输入样本到输出二维平面的有序特征映射。图 1 表示出这种自组织映射网络。



(a) 网络拓扑结构



(b) $N_j(t)$ 形状变化

图 1 自组织映射网络

设输入层有 n 个神经元,它们分别有输入 $x_1(t), x_2(t), \dots, x_n(t)$ 。输入层每一个神经元,通过权值 $\bar{W} = [w_{ij}]$ 与输出层的每一个神经元相连。输出层神经元按二维形式排列,邻近的神经元有侧向连接。

在输出层的竞争学习算法是这样安排的:对于获胜的那个神经元 j ,在其周围区域 N_j 范围内的神经元在不同程度上得到兴奋,而在 N_j 范围外的神经元均被抑制。这个 N_j 区域的形状可以是正方形或多边形,但其面积随着时间 t 的增加而逐渐缩小到

只包括一个或少数几个神经元,这些神经元就反映出一类输入样本的特征属性。

Kohonen 自组织特征映射算法如下:

(1) 网络初始化 设输入层和输出层分别有 n 个和 m 个结点, $w_{ij}(t)$ ($1 \leq i \leq n, 1 \leq j \leq m$) 表示输入 i 至输出层结点 j 的权值。赋予 $w_{ij}(0)$ 以小的随机数值。将任意一输出结点 j 的初始输出邻域 $N_j(0)$ 设得稍大一点;

(2) 输入样本组 设共有 p 个样本组,将其中的一组样本 $X^{(k)}(t) = [x_1^{(k)}(t), x_2^{(k)}(t), \dots, x_n^{(k)}(t)]$, $k=1, 2, \dots, p$ 加在输出结点上;

(3) 计算距离 将输出结点 j 与某样本组间的距离 d_j 按

$$d_j = \sum_{i=1}^n [x_i^{(k)}(t) - w_{ij}(t)]^2, j = 1, 2, \dots, m \quad (1)$$

计算出来;

(4) 选择最小距离 将具有最小 d_j 值的输出结点 j^* 选出,即

$$j^* = \{j \mid d_j = \min\{d_j\}, j = 1, 2, \dots, m\} \quad (2)$$

(5) 调整权值 对 t 时刻的输出结点 j^* 及其邻域 $N_{j^*}(t)$ 内的结点权值进行调整,即

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)[x_i^{(k)}(t) - w_{ij}(t)] \quad 1 \leq i \leq n \quad (3)$$

增益系数 $\eta(t)$ ($0 < \eta(t) < 1$) 随着时间逐渐减小,从而放慢权值的调整速度。邻域 $N_{j^*}(t)$ 也随时间逐渐减小面积,以最大限度突出表征该样本组的输出层的特定区域;

(6) 转步骤(2) 输入另一组样本,重复以上步骤。

Kohonen 将这种自组织映射方法应用于语音识别,他构造了一种基于神经网络原理的声控打字机。按自组织映射方式工作的神经网络,实现芬兰语的音素分类,其语音识别率为 93% ~ 98%。

参考文献

1. Kohonen T. Self-Organized Formation of Topologically Correct Feature Maps. Biol. Cybern., 1982, 43:59 ~ 69

2. 靳蕃,范俊波,谭永东. 神经网络与神经计算机原理·应用. 成都:西南交通大学出版社,1991

(靳蕃)

zonghe lilun xingneng

综合理论性能 (composite theoretical per-

formance, CTP) 美国政府为限制较高性能计算机出口所设置的运算部件综合性能估算方法。CTP 以百万次理论运算每秒 MTOPS 表示。CTP 自 1991 年 9 月 1 日起,启用根据估算结果及所定 CTP 数值限额,确定出口许可。停止使用原来设置的数据处理速率 PDR 值估算方法。

CTP 的估算方法为首先算出处理部件每一计算单元(如算术逻辑部件)的有效计算率 R ,再按不同字长加以调整,得出该计算单元的理论性能 TP ,所有组成该处理部件配置的计算单元 TP 的总和即为综合理论性能 CTP 。

例如 定点加法单元的 $R = \frac{1}{3 \times t_{\text{定点}+}}, t_{\text{定点}+}$ 为定点加法执行时间(μs);

定点乘法单元的 $R = \frac{1}{t_{\text{定点} \times}}, t_{\text{定点} \times}$ 为定点乘法执行时间(μs);

浮点+, × 单元的 $R = \frac{1}{t_{\text{浮}+}}, \frac{1}{t_{\text{浮} \times}}$ 等, $t_{\text{浮}+}, t_{\text{浮} \times}$ 分别为浮点加及乘的执行时间(μs)。

按操作数字长对 TP 加以调整:

$$TP = RL$$

$$L = (1/3 + WL/96)$$

式中 WL 为字长(b)。

对单个计算单元的处理部件:

$$CTP = TP$$

对由 n 个计算单元组成的多计算单元的处理部件:

$$CTP = TP_1 + c_2 \times TP_2 + \dots + c_n \times TP_n$$

TP_1 为 n 个 TP 中的最高值。对共享存储的多计算单元的处理部件,其

$$c_2 = c_3 = \dots = c_n = 0.75$$

c_i 为计算单元间互联强度系数。 (詹文岛)

zonghe yewu shuzi wang

综合业务数字网 (integrated services digital network, ISDN) 提供端到端的数字连接及标准的多用途的用户网络接口,以支持包括电话和非电话多种服务的一种通信网。它是以综合数字网 (IDN) 为基础发展而成的。用户通过一组多用途的用户网络接口接入网内。

业已存在的各种通信网如电话网、用户电报网、数据通信网等都是各自独立的。这种以不同的通信业务组网的方式存在着许多缺点,如每种网络都需

要专门的物理连接、专门的终端、不同的用户网络接口、不同的接入过程、不同的寻址过程和单独的号码簿以及单独的运营管理部门等。这些无论对于用户还是管理部门,在投资、效率、方便使用以及运行管理和引入新的业务等方面都有很大的问题。为了克服上述缺点,从根本上改变不同业务网络之间的隔离状况,需要用单一的网络来提供各种不同类型的业务,这就是综合业务数字网(ISDN)。用户只要通过 ISDN 网络终端上的标准用户网络接口,利用一条用户线就能实现各种通信业务。

以国际电报电话咨询委员会(CCITT)为中心,各国对 ISDN 进行了大量的研究。1984 年 CCITT 制定了 ISDN 的 I 系列建议第一版,提出了对于 ISDN 的基本观点。1988 年 CCITT 总结归纳了各国的研究成果,对 ISDN 技术规范作了详细的规定。自此,ISDN 开始迅速发展。

ISDN 的发展大致可以分为以下 4 个阶段:

(1) 实现 64 kb/s 电路交换(1985 年—1990 年);

(2) 实现具有电路交换和分组交换功能的 ISDN(1990 年—1995 年);

(3) 实现具有各种智能功能的 ISDN(1995 年—2000 年);

(4) 实现宽带 ISDN(1996 年起),即 B-ISDN。

在 ISDN 技术中,通常将只能提供一次群速率(即 1.5~2 Mb/s)以内业务的 ISDN 称为窄带 ISDN(NISDN)。为适应诸如电视会议、广播电视、高清晰度活动图像和高速数据等高速宽带通信业务的需求,CCITT 着手制定基于异步传送模式(ATM)的宽带 ISDN(BISDN)的技术标准。在 1990 年通过了 BISDN 的第一组建议,在 1992 年又通过了一些新的建议。

ISDN 的国际标准

CCITT 的 I 系列建议包含下列 6 个主要部分,它们是研究和实现 ISDN 的依据。

(1) I.100 系列 总体(ISDN 的基本概念、建议的结构、术语、一般方法);

(2) I.200 系列 业务特点(承载业务、用户终端业务);

(3) I.300 系列 网络结构及功能(ISDN 功能原理、基准(参考)模型、寻址、路由、连接类型和性能);

(4) I.400 系列 用户网络接口(基本接口和一次群速率的 1 层~3 层,复用、速率适配以及对现

有接口的支持);

(5) I.500 系列 网间互连接口;

(6) I.600 系列 维护原则。

此外还有 G 系列、M 系列和 Q 系列。它们仅与 ISDN 网络的内部结构有关。

ISDN 业务

ISDN 的根本任务是向用户提供电话的和各种非电话的通信服务,这些服务总称为 ISDN 业务。通常它可以分为承载业务、用户终端业务和附加业务 3 类。

(1) 承载业务是一种单纯的信息传送业务。它由网络提供,其任务是将信息由一个地方传送到另一个地方而不作任何处理。这类业务包含了开放系统互连基准(参考)模型 1 层~3 层的功能,在 CCITT I.230, I.231, I.232 的系列建议中定义了承载业务。

(2) 用户终端业务是面向用户的通信或信息处理的业务。它由网络和终端设备共同提供,包含开放系统互连基准(参考)模型 1 层~7 层的全部功能。在 CCITT 的 I.241 建议中定义了用户终端业务。

(3) 附加业务是在承载业务和用户终端业务基础上附加的业务。它的目的是为了给用户提供更多更方便的服务。在 CCITT 的 I.250, I.251~I.257 建议中规定了附加业务。

宽带 ISDN(BISDN)

宽带 ISDN 具有以下特点:

(1) 在宽带用户网络接口上能够提供 155.220 Mb/s 全双工访问方式以及 155.220 Mb/s 输入、622.080 Mb/s 输出、622.220 Mb/s 全双工三种访问方式(参见串行传输);

(2) 能够提供各种连接形态,如点对多点等;

(3) 信息传送的延迟及其变化很小,信息传送的损失也很小;

(4) 不仅能支持传送图像、语音(言语)等固定比特率型的业务,还能有效地支持分组通信等可变比特率型的业务。

在宽带 ISDN 中,采用统一的传输与交换技术,即异步传送模式(ATM),并且用光缆代替金属线和通信电缆作为用户线。

参考文献

1. 李津生,秋山稔.综合业务数字网与异步转移模式 ISDN & ATM.合肥:中国科学技术大学出版社,1993

2. 胡道元. 计算机网络工程指南. 北京: 电子工业出版社, 1993

3. 程时端. 综合业务数字网. 北京: 人民邮电出版社, 1993

4. Behrouz Forouzan etc. Introduction to data Communications and Networking. WCB. McGraw-Hill 1998

5. 潘乞, 朱丹宇, 周正康译. 数据通信与网络. 北京: 机械工业出版社, 2000 (史美林)

zongxian biao zhun

总线标准 (bus standard) 计算机系统中各个模块间互连的标准界面。该界面对系统中各模块而言都是透明的, 界面的任一方只需根据总线标准的要求来实现接口的功能, 而不必考虑另一方的接口方式。按总线标准设计的接口是通用接口。采用总线标准可以为计算机接口的软硬件设计提供方便。对硬件设计而言, 由于总线标准的引入, 使各个模块的接口芯片的设计相对独立。对软件设计而言, 总线接口的引入也给接口软件的模块化设计带来了方便。

总线标准必须有详细和明确的规范说明, 一般包括如下几个部分: ①机械结构规范, 确定模板尺寸、总线插头、边沿连接器等的规范及位置; ②电气规范, 规定信号工作时的高低电平、动态转换时间、负载能力以及最大额定值; ③功能规范, 确定各引脚信号的名称、定义、功能与逻辑关系, 对相互作用的协议(定时)进行说明。

总线标准的制订通常有两种途径: ①某计算机公司(或生产厂)在发展自己的计算机系统时所采用的一种总线, 由于性能与价格等方面的优势, 得到 OEM(原始设备制造厂)的普遍接受, 按此总线规范开发相应的配套产品, 进而形成一种为国际工业界广泛支持的实用总线标准; ②在标准化组织的主持下由专家小组制订总线标准, 标准推出后即可由厂家和用户使用。

从事接纳和主持制订总线标准工作的有 IEEE(美国电气与电子工程师协会), IEC(国际电工委员会)和 ANSI(美国国家标准局)组织的专门标准化委员会, 这些委员会一方面为适应不同应用水平的要求从事开发和制定总线标准或建议草案; 另一方面对现有的由一些公司提出的并为国际工业界广泛支持的实用总线标准进行遴选、研究、修改和评价, 给以统一编号, 作为对该总线标准的认可。

随着计算机系统的发展, 总线在不断发展和完善, 一些老的总线标准已不适应技术发展的需要, 因而有的被淘汰, 如 S-100 总线; 有的被改进和完善, 如 STD 总线。而新的总线标准也在不断产生, 近 20 年来比较流行的系统总线有 IBM PC/XT 总线、ISA 总线、EISA 总线、VME 和 VME64 总线、MCA 总线、FutureBus 和 FutureBus + 总线、STD 和 STE 总线、VL-Bus、PCI 和 PCI-X 总线、NGIO、InfiniBand 等。其中, **IBM PC/XT 总线** 是早期的计算机总线, 是以 Intel 8088 微处理器为基础设计的, 有 62 条信号线, 其中 8 条数据线、20 条地址线。它采用大主板上带有扩展输入输出槽的结构。此外影响较大的有:

(1) **ISA 总线** 最早用于以 80286 微处理器为基础的 IBM PC/AT 机中, 又称 AT 总线。它是在 IBM PC/XT 总线基础上增加 1 个 36 引脚的扩展槽, 其数据总线宽度为 16 位, 工作频率 8 MHz, 数据传输速率最高为 16 MB/s。286, 386 和 486 微型计算机大多采用 ISA 总线, Pentium 系列微型计算机中也有同时采用 ISA 总线和 PCI 总线以连接不同速度的部件。

(2) **EISA 总线** ISA 总线的扩展。EISA 是 1988 年由 9 家计算机工业公司联合推出的。它与 ISA 总线完全兼容, 并支持多个总线主控器, 加强了 DMA 功能, 增强了突发式传输, 数据总线宽度增为 32 位, 工作频率 8 MHz, 最大的数据传输速率为 33 MB/s。最多可支持 6 个总线主控器, 是一种支持多处理器的高性能 32 位总线标准。

(3) **Futurebus 与 Futurebus +** 它们是由 IEEE 开发的高性能总线标准, 能在不同的多处理器系统中应用。其中 Futurebus + 是 Futurebus 标准的扩展, 其差别的关键点有 3 个: ①Futurebus 支持 32 位数据总线; Futurebus + 支持的数据总线宽度为 32 位、64 位、128 位和 256 位。②Futurebus 只支持分布仲裁协议, 而 Futurebus + 支持分布和集中式两种模式。③Futurebus + 包含 3 位能力域, 它允许模块声明它的能力, 用以调节总线交换的重要模式。

(4) **PCI 与 PCI-X 总线** 该类总线是一种高性能的局部总线, 可配置成 32 位或 64 位总线, 工作频率可以是 33 MHz 或 66 MHz。PCI 总线有严格的规范, 保证了良好的兼容性。PCI-X 总线则是由很多公司提出的一种新的 I/O 接口和总线标准。PCI-X 总线可支持 66/100/133 MHz 的工作频率, 在 133 MHz 时, 64 位总线的最大带宽达 1 066 MB/s。PCI 与 PCI-X 总线由于其优秀的性能、良好的兼容性和

易用性,已经成为当前最为流行的计算机系统总线标准。

(5) InfiniBand 总线 是 NGIO 和 Futurebus I/O 技术的融合,采用全新的体系结构,与传统的 PCI 无法兼容。在 InfiniBand 系统中,远程存储器,网络和服务端之间的连接是通过一条位于中心的 InfiniBand 控制芯片和中继线完成的。采用 InfiniBand 通道设计,外部设备可以离服务器远达 10 km。根据不同需要,InfiniBand 标准为通道适配器设置三种工作方式,分别提供 500 MB/s, 2 GB/s, 6 GB/s 的带宽。InfiniBand 总线主要用于服务器系统中。使用 InfiniBand 总线的系统将会得到更高的带宽和扩展能力,增强系统的灵活性。(孙德文)

zongxian kongzhiqi

总线控制器 (bus controller) 计算机系统中用以代替中央处理器 (CPU), 提供总线控制和命令信号的一种专用集成电路。一般情况下, CPU 输出表征该总线周期要存取的设备的状态信号, 通过总线控制器产生该总线周期所需的全部总线控制信号及命令输出信号, 并对连在总线上的存储器和输入输出 (I/O) 设备进行控制。总线控制器也称**系统总线控制器**。

总线控制器的内部结构主要由状态解码、控制输入逻辑、状态机、控制输出逻辑和命令输出逻辑等组成, 其电路结构如图 1 所示。控制输出提供允许地址锁存 ALE、数据收发 DT/ \bar{R} 、数据允许 DEN 等信号, 并用这些信号对存储器和输入输出系统的地址锁存、数据收发、允许写和允许输出等进行控制。

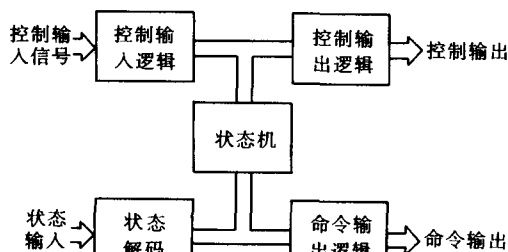


图 1 总线控制器电路结构

使用总线控制器的数据允许 DEN 和数据收发 DT/ \bar{R} 信号控制数据总线收发器; 其中, 用 DEN 启动数据收发器, 用 DT/ \bar{R} 控制数据收发器的方向。利用对 DEN 和 DT/ \bar{R} 的定时控制, 可以避免总线主

控设备、数据总线收发器等总线争用。

利用允许地址锁存 ALE 的输出确定对地址锁存的时间。从前一个总线操作完成到下一个总线操作出现在锁存器输出端为止, ALE 至少要提供一个系统地址保持时间。利用该保持时间, 支持多总线和公用存储系统。另外, 系统命令的延迟也受到总线控制器的控制。命令延迟信号允许增加地址或写数据的时间, 以便系统总线命令借助延迟完成各种总线操作。

使用总线控制器, 是为了解决 CPU 控制负载过重与外引线数目受限制等问题。它使 CPU 仅产生状态信号, 不直接产生控制存储器或 I/O 电路的读写信号。在构成系统时, 就需要总线控制器译码状态信息, 从而产生能直接控制驱动存储器和 I/O 读写的信号。由于总线控制器需要较高的负载能力, 因此, 一般采用双极型电路。

典型的总线控制器产品为 82 C 288, 它采用高速 CMOS 技术。其主要功能有: 提供局部总线和系统总线命令和控制, 单 +5V 电压, 全静态器件, 与 HMOS 技术的 82288 全兼容。

参考文献

1. John N. 32-bit Microprocessor. Collins Professional and Technical Books, 1986
2. 艾德才. 长城 286 386 体系结构与汇编程序设计. 天津: 天津大学出版社, 1988 (孙育宁)

zongxian zhongcai qi

总线仲裁器 (bus arbiter) 在总线结构的计算机或总线结构的多处理机系统中, 对多个设备使用系统总线请求进行仲裁的集成电路。得到总线控制权的设备称为总线主控制器。总线仲裁器与**总线控制器**一起构成总线上设备的接口。

总线仲裁器的内部结构主要由仲裁机构、状态发生器、多总线接口、本地总线接口和控制电路组成, 典型电路结构如图 1 所示。

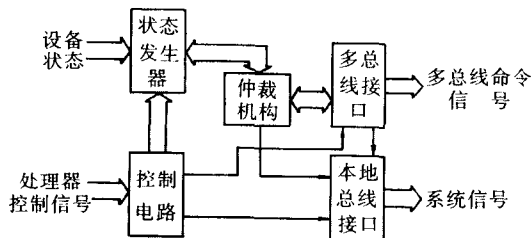


图 1 总线仲裁器电路结构

从原理上讲,每个系统总线都有一个总线仲裁器。但也有些总线不需要仲裁,这类总线称为自仲裁总线。自仲裁总线没有总线仲裁器。使用系统总线的设备可以是中央处理器(CPU)或输入输出设备,这些设备都可以成为总线主控制器,控制系统总线。

对于总线仲裁,一般采用三种方式:菊链式、独立请求—允许线式和轮询式。菊链式的工作原理是:首先,将需要总线的设备的总线请求置1,向总线仲裁器提出请求;总线仲裁器监视总线“忙”线,当总线不忙时,总线仲裁器检查是否有总线请求信号;若有请求信号,则总线仲裁器发送总线允许信号,总线允许信号沿总线设备一级一级向下传递,直到请求总线的设备为止。当设备收到总线允许信号后,禁止该允许信号继续向下传递,并置总线“忙”信号,通知总线仲裁器总线在使用;然后,该设备使用总线传送信息;当信息传送完毕时,设备将总线“忙”信号清零,总线仲裁器又开始新一轮查询。

独立请求—允许线式的工作原理是:为总线上的每个设备设置独立的总线请求线和总线允许线。当设备需要总线时,通过独立的总线请求线向总线仲裁器提出请求。总线仲裁器对总线请求进行优先级判断。然后,通过独立的总线允许线把总线允许信号发送给请求中的优先级最高的设备,使该设备获得总线使用权,并由该设备给总线“忙”信号置位,通知总线仲裁器总线在使用。当信息传送完毕后,设备将总线“忙”信号清零,仲裁器又开始新一轮查询。

轮询式的工作原理是:当总线仲裁器接收到总线请求信号后,以一种特定的次序扫描总线设备,直到找到需要的设备为止。其余过程与独立请求—允许线式类似。

以上三种方式各有特点。菊链式控制简单,但信号延迟时间长,设备优先级是根据设备距总线仲裁器的远近来确定的。这种方式可能出现优先级低的设备长时间得不到总线请求响应的情形。独立请求—允许线式可以缩短响应时间,改变设备优先级顺序,但增加了控制线的条数。轮询方式是以上两种方式的折衷。

与总线仲裁器配合使用的还有总线控制器、数据收发器和地址锁存器等。

典型的总线仲裁器有美国 Intel 公司的 8289,它采用双极型工艺。其主要功能有:提供多机系统总线协议、总线同步,提供与 8288 总线控制器的接口,

与 MULTIBUS 标准兼容等。

参考文献

1. Wilkinson B. Computer Architecture Design and Performance. New York: Prentice Hall, 1991
2. Baron R J, Highbie L. Computer Architecture. Addison - Wesley Publishing Company, 1992

(孙育宁)

zuyong xianlu wang

租用线路网 (leased line network) 由租用线路连接而成的一种通信网。又称专线网。在频繁进行电话通信或数据通信的站点之间租用专线比拨号电话线更为可取。它具有效益价格比高、使用方便、专用与安全可靠等优点。

租用线路,也称专用电路,是在两个固定的站点之间的一条专门的链路,通信双方可随时使用。租用线路通常以一些固定的传送速率工作,如 9.6 kb/s, 19.2 kb/s, 56 kb/s, 64 kb/s 等。在传送速率要求更高的情况下,可租用 T1 (1.544 Mb/s, 美国标准) 或 E1 (2.048 Mb/s, 欧洲标准)。

租用线路网构成方式有如下几种:

(1) 直通专线方式 用点对点的直通专线信道连接终端、计算机与计算中心的主机。这是一种基本方式,每一终端、计算机独自占用一条信道,适用于传输大数据量的业务。

(2) 分支专线方式 它又可分为线路分支专线和调制解调器分支专线两种方式。这种方式可提高线路利用率,适用于终端数据量相对较小的业务。

(3) 频分复用(FDM)方式 在专线两端加上频分复用设备,如载波机、载报机等来提高线路利用率(参见**频分多路复用**)。

(4) 时分复用(TDM)方式 它是按时间片进行多路复用以提高线路利用率(参见**时分多路复用**)。

(5) 信息集中方式 采用信息集中器、终端集中器、统计复用器等设备先把信息集中,再行传送,不仅能起到信息集中的作用,还可进行速率、电码以至规程变换等。

(6) 线路集中方式 除了有报路集中和话路集中两种传统的方式外,还普遍采用各种类型的**集线器**,既可实现**局域网**的互连,也可通过专线进行远程连网。

(7) 混合方式 为求技术先进和经济合理,租用线路网往往采用多种方式混合构成。(史美林)

zuhe luoji

组合逻辑 (combinatory logic) 研究组合子的形式描述和性质的逻辑。大约从 1920 年开始, M. Schönfinkel 和 H. Curry 先后独立地发明了组合子, 以后逐渐建立了关于组合子的形式系统, 即**组合逻辑系统**。它原是作为高阶逻辑的子部分引入的, 并试图作为无类型数学的一个基础。

组合逻辑系统的字母表由可数无穷多个变元以及常元 K 和 S (称为组合子) 组成。它的项归纳定义如下: (1) 变元和常元是项; (2) 若 X 和 Y 都是项, 则 (XY) 是项; (3) 每个项都可以通过有穷次应用 (1) 和 (2) 得到。不含变元的项称为组合子。如 $SKK, S(KS)K, S(BBS) (KK)$ 和 $SS(KI)$ 等都是组合子。项之间的弱化归关系 \triangleright_{ω} 和弱相等关系 $=_{\omega}$ 是组合逻辑系统中两个重要的基本概念。它们是分别由两组不同的公理和规则归纳定义的。组合逻辑系统的表达能力很强, 可以在系统中定义自然数, 进而表达递归函数, 而且组合逻辑系统可定义的函数类与递归函数类相同, 从而获得了计算的又一模型。

组合逻辑与程序设计语言密切相关。它其实就是一种抽象的程序设计语言, 这可从 20 世纪 50 年代的 LISP 语言及后来的函数式语言看出。D. Scott 关于 λ -演算, 和组合逻辑的模型论, 对计算机语言的形式语义学产生了重大影响。组合逻辑与 λ -演算关系紧密, 它们几乎具有相同的能力

参考文献

1. Hindley J R, et al. Introduction to Combinators and λ -Calculus. Cambridge University Press, 1986
2. Curry H B, et al. Combinator Logic. I. II. North Holland, 1985, 1972 (宋方敏)

zuhe suanfa

组合算法 (combinatorial algorithm) 计算对象是离散的、有限的数学结构的组合学问题的算法。组合算法的用途十分广泛。从方法学的角度, 组合算法包括算法设计和算法分析两个方面。关于算法设计, 已经总结出若干带有普遍意义的方法和技术, 包括动态规划、回溯法、分枝限界法、分治法、贪心法等。尽管如此, 组合算法的设计仍然是一门艺术, 需要高度的技巧和灵感。算法分析的任务是分析算法的优劣, 主要是讨论算法的时间复杂性和空间复杂性。它的理论基础是组合分析, 包括计数和枚举。计算复杂性理论, 特别是 NP 完全性理论,

与组合算法是紧密相关的。NP 完全性概念的提出, 正是为了刻画包括旅行商问题、图着色问题、整数规划等在内的一大批组合问题的计算难度。计算复杂性理论研究算法在时间和空间限制下的能力以及问题的难度, 使组合算法的研究有了更加清晰的框架, 将组合算法的研究提高到一个新水平。

单纯形法 G. B. Dantzig 在 1947 年提出的一种线性规划算法, 他本人以及其他学者后来又提出多种形式的变形和改进。实践表明, 单纯形法及其变形和改进是非常行之有效的, 在市场上已经形成许多可以有效解决大型线性规划问题的软件包。线性规划研究线性目标函数在一组线性等式与线性不等式约束下的极值问题。这本来是连续问题, Dantzig 发现线性规划问题的可行解集 (即满足约束条件的点的全体) 是一个超多面体。如果它的最优解存在, 那么最优解一定可以在这个超多面体的某个顶点取到。由于超多面体的顶点只有有限个, 从而使线性规划成为一个组合优化问题。单纯形法是按照一定的规划, 从可行解集的一个顶点转移到另一个顶点, 使得目标函数的值不断地得到改进, 最后达到最优。尽管单纯形法一直使用得很好, 但在最坏情况下它需要指数运行时间, 从而使线性规划问题是否属于 P 类一度成为人们关心的问题。1979 年前苏联数学家 Л. Г. Хачиян 提出一个多项式时间的线性规划算法——椭球算法, 从而解决了这个问题。1984 年印度数学家 N. Karmarkar 又提出一个新的更好的多项式时间算法——投影算法。

排序和检索 将给定的元素序列按照某种顺序关系重新排列成有序序列称作排序。例如将 n 个数组成的序列按照从小到大的顺序重新排列; 将 n 个英语单词组成的序列按照字典顺序重新排列。在给定的集合中查找某个特定的元素称作检索。例如从给定的 n 个数中找到最大的数。排序和检索算法已经成为数据结构中不可缺少的部分, 是计算机科学技术中最基本、使用最频繁的算法。正因为如此, 它们也是研究得最细致的一类组合算法 (参见**排序算法**)。

图与网络优化算法 组合算法中内容最丰富的部分。图论中的计算问题包括图的搜索、路径问题、连通性问题、可平面性检验、着色问题、网络优化等。图论中的著名算法有求最小生成树的 Kruskal 算法、求最短路的 Dijkstra 算法和 Floyd 算法、求二部图最大匹配 (指派问题) 的匈牙利算法、求一般图最大匹配的 Edmonds “花” 算法、求网络最大流和最小割的

标号法等。

贪心法与拟阵 贪心法是求解关于独立系统组合优化问题的一种简单算法,求最小生成树的Kruskal算法就是一种贪心法。但是,贪心法并不总能找到最优独立集。贪心法能求得最优独立集的充分必要条件是 L 为一个拟阵。事实上,求最大生成树是关于拟阵的组合优化问题,而二部图的所有匹配构成的独立系统 U 不是拟阵。

穷举搜索 组合算法要解决的问题只有有限种可能,在没有更好办法时总可以用穷举搜索的办法来解决,即逐个检查所有可能的情况。当情况较多时这样做是很费时的。实际上,并不需要机械地检查每一种情况,常常有可能提前判断出某些情况不可能取到最优解,从而可以提前舍弃这些情况。这样便“隐含地”检查了所有情况,既减少了搜索量,又保证不漏掉最优解。参见算法设计条目中回溯法。

分支限界法 一种用于求解组合优化问题的排除非解的搜索方法。它的基本思想是:把问题分成若干个子问题,估计子问题的目标函数值的上界或下界。对于最大值问题,子问题的下界也是原问题的下界。当子问题的上界小于原问题的下界时,不可能在这个子问题中取得原问题的最优解,舍去这个子问题。否则将这个子问题再划分成若干更小的子问题,重复上述过程,直到没有需要检查的子问题为止。

其他组合算法还有动态规划,快速傅里叶变换等。

参考文献

1. Reingold E M, Nievegelt J, Deo N. Combinatorial Algorithms: Theory and Practice. Englewood Cliffs: Prentice Hall, 1977
2. Papadimitriou C H, Steiglitz K 著. 组合最优化算法和复杂性. 刘振宏, 蔡茂诚译. 北京: 清华大学出版社, 1988
3. 卢开澄. 组合数学: 算法与分析, 上下册. 北京: 清华大学出版社, 1983 (张立昂)

zuhexue

组合学 (combinatorics) 研究满足各种附加条件的有限个对象的集合(称为组态)的数学分支。又称组合数学、组合理论或组合分析。组合学研究的问题主要有:计数问题,即要求得到组态或组态等价类的个数;存在性问题,即希望判明某种特定组态

是否存在,其中特别希望得到所论组态存在的简明充分必要条件;枚举、构造和算法问题,即具体列出要求的全部组态和给出得到这些组态的算法;优化问题,即根据某些明确的标准找出最优或近乎最优的组态。当然,为解决上述各类问题必须深入研究已知或未知组态的结构和内在性质。由于在数学、自然科学、管理科学的很多分支,尤其在计算机科学的理论和应用上经常要处理形形色色的组态,因此,它是近几十年来增长最快的数学学科之一。但就其理论现状而言,组合学尚未显示出其统一性。目前在组合学中已形成了在对象和术语上自成体系的几个部分,最大同时也是与计算机科学联系最紧密的部分是图论,此外还有组合计数、组合设计、组合最优化和组合几何等部分。

组合学是一个古老而又年轻的数学分支。据传大禹治水时(约公元前2200年)曾见到“神龟”背上一种图案,用阿拉伯数字表示就是下列3阶幻方

$$\begin{array}{ccc} 4 & 9 & 2 \\ 3 & 5 & 7 \\ 8 & 1 & 6 \end{array}$$

它的每行、每列以及二对角线上数字之和都是15。这一历史上最早出现的非平凡组态至少在东汉末年(公元200年前)已有确切记载。在11、12世纪,中国、波斯、印度的学者各自得到组合数(即 n 元集中 r 元子集的个数)的公式,其中宋代杨辉在《详解九章算法》(1261年)中载有11世纪人贾宪的“开方法本源”图,即表示组合数间递推关系的三角形图表。17世纪以来,组合学逐渐成为一个独立的数学分支,这主要归功于欧洲的很多数学家在解决概率计算、正整数分析等数学问题和一些智力难题时的大量研究工作。其中德国数学家、哲学家L. W. 莱布尼兹在1666年出版了组合学的第一部著作《论组合的艺术》,他希望这一学科能应用到“所有科学领域”,近代数学意义下的“组合”一词也源出于此。到20世纪30年代,组合学渐趋深入,一些专著相继问世,也得到了一批一般性的定理。60年代以来,以《组合理论杂志》在1966年刊行为标志,组合学进入蓬勃发展的新阶段。计算机技术的急剧发展和信息时代的到来,已经并将继续提出大量富有理论和实际意义的组合学课题,它们是推动组合学不断深入发展的巨大动力。

因为形形色色的组态无所不在,所以组合学的内容从来就很驳杂。这里简单介绍它的几个主要部分和两个有广泛影响的存在性定理。

图论 这是组合学中最大的一部分,有的文献把它当作与组合学并列的数学分支,其内容请参看图论条目。这里仅指出一点:大规模并行计算机系统的互联网络通常以有向或无向图为数学模型,称为网络拓扑,这时图的结点对应于处理机,图的边则对应于处理机间的通信链路,于是图论的研究与互联网络的设计和分析密切相关。

组合计数 其目标是确定一类组态中组态的个数。最基本的是排列与组合的计数公式:元素取自 n 元集 S 的不同的 k 个元的序列,个数是 $(n)_k = n(n-1)\cdots(n-k+1)$, 不一定不同的 k 个元的序列个数是 n^k ; S 中 k 个不同元组成的子集个数是 $\binom{n}{k} = (n)_k / k!$, k 个不一定不同元组成的多重子集个数是 $\binom{n+k-1}{k}$ 。另一经典计数问题是求一个 n 元集分拆成两两不交的 k 个非空子集之并的分拆方式数 $S(n, k)$, 称为第二类斯特林数, 它有表示式 $S(n, k) = \frac{1}{k!} \sum_{i=0}^k (-1)^i \binom{k}{i} (k-i)^n$ 。一般来说,很难求得用初等函数表示的计数公式,但已经发展了一些研究计数问题的方法,主要有:

(1) 利用生成函数 这是研究计数问题的主要途径,其基本思想是:为了获得有关一个有限或无限数列 $\{a_k: k \geq 0\} = a_0, a_1, a_2, \dots$ 的知识,用一个形式幂级数 $g(x) = a_0 + a_1x + a_2x^2 + \dots$ 来整体地表示这个数列,称 $g(x)$ 为该数列的生成函数(也称母函数或发生函数)。再通过 $g(x)$ 和对它的运算得到原数列的性质,而数列的一般项正是计数问题的解。如规定 $\binom{n}{0} = 1, k > n$ 时 $\binom{n}{k} = 0$, 则数列 $\left\{ \binom{n}{k}: k \geq 0 \right\}$ 的生成函数是 $(1+x)^n$, 由此 $\binom{n}{k}$ 被称为二项式系数。令 $x=1$ 即得恒等式 $\sum_{k=0}^n \binom{n}{k} = 2^n$, 令 $x=-1$ 得 $\sum_{k=0}^n (-1)^k \binom{n}{k} = 0$, 还可以利用此生成函数推得不少关于二项式系数的恒等式。

(2) 利用递归关系 即对数列 a_0, a_1, a_2, \dots 找到一种规则,使得从某一项起,数列的每一项可通过此规则由其前面的那些项惟一确定。例如,在各种问题中出现的斐波那契数列 $\{f_k: k \geq 0\}$ 即可由递归关系 $f_k = f_{k-1} + f_{k-2} (k \geq 2)$ 及初始值 $f_0 = f_1 = 1$ 完全确定。通过它可以求得此数列的生成函数 $\sum_{k \geq 0} f_k x^k =$

$$4/(1-x-x^2), \text{ 从而 } f_k = \frac{1}{\sqrt{5}}(\alpha^{k+1} - \beta^{k+1}), \text{ 这里 } \alpha = \frac{1}{2}(1+\sqrt{5}), \beta = \frac{1}{2}(1-\sqrt{5}).$$

(3) 容斥原理和反演公式 已知有限集 A 以及 A 的 n 个子集 A_1, A_2, \dots, A_n , 则 A 中不属于任一子集 $A_i (i=1, 2, \dots, n)$ 的元素个数是 $|A| - \sum_i |A_i| + \sum_{i < j} |A_i \cap A_j| - \sum_{i < j < k} |A_i \cap A_j \cap A_k| + \dots + (-1)^n |A_1 \cap A_2 \cap \dots \cap A_n|$, 这里 $|S|$ 表示有限集 S 的元素个数。这就是最简单形式的容斥原理(也称筛式或逐步淘汰原理),用它以求得一些计数公式。如错位排列,即 $1, 2, \dots, n$ 的共 $n!$ 个全排列中使得每个 $i (i=1, 2, \dots, n)$ 都不排在第 i 位的排列个数是 $n! - \binom{n}{1}(n-1)! + \binom{n}{2}(n-2)! - \dots + (-1)^n \binom{n}{n}(n-n)! = n! \left(1 - \frac{1}{1!} + \frac{1}{2!} - \dots + \frac{(-1)^n}{n!} \right)$ 。上述容斥原理有诸多推广,其中最深刻的结果是 G. C. Rota 在 1964 年得到的局部有限半序集上的反演公式,它以容斥原理和数论上的默比乌斯反演公式为特例,以简明的方式统一了很多具体结论,是组合学的一个重大进展。

(4) 波利亚计数定理 有的问题中计数的对象不是组态,而是组态的等价类。例如,分别用红、蓝二种颜色来染一个可任意活动的立方体的 6 个面时,有理由认为恰有一面染成红色的 6 种染法是等价的,因为把立方体作一旋转就能把红面转成顶面。如果一种染法能通过立方体的适当旋转等同于另一种,则认为这二种染法是等价的。所以不等价的染法不是 2^6 种,而可以穷举所有情况得知共有 10 种。G. Pólya 在 1937 年得到的计数定理是迄今为止关于等价类计数的主要理论基础,他本人后来把这一定理的内容叫做“相对于给定置换群的不等价组态的计数”。对立方体染色的例子来说,如用 x, y 代表红、蓝色,则从波利亚定理可求得生成函数 $x^6 + x^5y + 2x^4y^2 + 2x^3y^3 + 2x^2y^4 + xy^5 + y^6$, 它展示了把 6, 5, 4, 3, 2, 1 和 0 个面染成红色的不等价染法分别有 1, 1, 2, 2, 2, 1 和 1 种,总的等价染法共 10 种。

(5) 渐近计数 绝大多数计数问题的解 a_n 的精确表示式无法求得或非常复杂,而在有的问题中人们主要关注当 n 趋于无穷时 a_n 的渐近性状。例如在算法分析中,人们更关注运算次数 a_n 当 n 趋于

无穷时是按线性方式 cn , 还是按 $cn^{\frac{3}{2}}$ 或 $cn^{\frac{1}{2}} \log n$ 方式增长? 这时我们希望得到 a_n 的渐近式 $a_n \sim b_n$, 其中 b_n 是比较简单的初等函数, 且有 $\lim_{n \rightarrow \infty} a_n / b_n = 1$ 。著名

的渐近式有斯特林公式 $n! \sim \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$, 这里的 e 是自然对数的底。和前面几种方法相比, 渐近计数主要依靠解析方法, 尤其是复变函数论。

组合设计 主要研究特定组态的存在性问题和构造方法。20 世纪中叶以来, 其内容日趋深广且渐成体系。下面是两种最有代表性的组合设计。

(1) 区组设计 最一般的情形可定义为 v 元集 X 以及 X 的 b 个子集 B_1, \dots, B_b 的族, 其中各子集 B_i 称为区组, b 个区组不一定不同。很重要的一类特殊区组设计是平衡不完全区组设计, 它进一步要求有正整数 k 和 λ , 使得每个区组都是 k 元子集, 且 X 的每一对不同元恰好同属于 λ 个区组。通常把这种设计称为 (v, k, λ) 设计, 并假设 $1 < k < v-1$ 。可证这时 X 的每一元恰好属于 $r = \lambda(v-1)/(k-1)$ 个区组, 而区组总数 $b = \lambda v(v-1)/(k(k-1))$ 。所以 (v, k, λ) 设计存在的必要条件是

$$\lambda(v-1) \equiv 0 \pmod{(k-1)}$$

$$\text{和 } \lambda v(v-1) \equiv 0 \pmod{k(k-1)}$$

但它们不是充分条件, 例如 $(15, 5, 2)$ 设计不存在。事实上, 确定能使 (v, k, λ) 设计存在的 v, k, λ 的值的精确范围一直是研究这种组态的中心问题。当 $k=3, \lambda=1$ 时, T. P. Kirkman 在 1847 年和 M. Reiss 在 1859 年独立地证明了 $(v, 3, 1)$ 设计 (也称为斯坦纳三元系) 存在的上述必要条件也是充分的。H. Hanani 在 1961 年进一步证明了当 $k=3, 4$ 时上述必要条件是充分的。对一般的 $k \geq 3$, R. M. Wilson 则证明了上述必要条件的“渐近充分性”, 即对任意给定的正整数 $k \geq 3$ 和 λ , 存在常数 $v_0 = v_0(k, \lambda)$, 使得当 $v \geq v_0$ 且 v, k, λ 满足上述必要条件时, 必定存在 (v, k, λ) 设计。

(2) 正交拉丁方 n 阶拉丁方可定义为数 $1, 2, \dots, n$ 排成的一个 $n \times n$ 矩阵, 其中每一行和每一列上的 n 个数互不相同。两个 n 阶拉丁方 $A = [a_{ij}]$ 和 $B = [b_{ij}]$ 称为正交的, 如果 n^2 个有序对 (a_{ij}, b_{ij}) ($i, j = 1, \dots, n$) 互不相同。欧拉在 1779 年提出这个概念, 并当 n 是奇数或 4 的倍数时构造出一对正交的 n 阶拉丁方, 但他认为当 $n = 4m + 2$ 时不存在一对正交的 n 阶拉丁方。直到 1900 年 G. Tarry 才通过穷举所有 6 阶拉丁方证明了欧拉的猜想当

$n=6$ 时为真。1958 年 R. C. Bose 和 S. S. Shrikhande, 以及稍后的 E. T. Parker 分别找到了一对正交的 22 阶和 10 阶拉丁方, 否定了欧拉猜想。接着三位数学家联手在 1959 年证明了一个令人惊异的结果: 除 $n=2$ 和 6 外, 必存在一对正交的 n 阶拉丁方。另外在 30 年代已证明对任一正整数 n 至多有 $n-1$ 个两两正交的 n 阶拉丁方, 且当 n 是素数幂时可具体给出 $n-1$ 个两两正交的 n 阶拉丁方。所以有待确定的最小阶是 $n=10$, 但至今人们尚未能找到三个两两正交的 10 阶拉丁方。

组合设计是在 20 世纪 20 年代数理统计中的试验设计和分析的推动下发展起来的, 它的实际背景不满足于证明特定设计的存在性, 还要求具体构造出这些组态。构造方法主要有借助于数论、代数、有限几何等数学工具以及计算机搜索的直接法和由小组态拼装成大组态的递归法。

两个存在性定理

(1) 霍尔定理 设 S_1, S_2, \dots, S_m 是有限集 S 的 m 个不一定不同的子集的序列, 如果 S 中有 m 个不同元 x_1, x_2, \dots, x_m 使 $x_i \in S_i$ ($i = 1, \dots, m$), 则序列 x_1, x_2, \dots, x_m 称为子集序列 S_1, S_2, \dots, S_m 的一个相异代表序列, 简记成 SDR。P. Hall 在 1935 年证明了下述有广泛影响的存在性定理: 有限集 S 的子集序列 S_1, S_2, \dots, S_m 具有 SDR 的充分必要条件是, 对任一正整数 $k \leq m$, 子集序列中任意 k 个子集的并集至少含有 k 个元。

(2) 拉姆齐定理 对任给的正整数 r, k 以及 $q_1, q_2, \dots, q_k \geq r$, 必存在数 N , 使当 $n \geq N$ 时, 对 n 元集 S 以及 S 的所有 r 元子集的集 $S^{(r)}$ 的任一有序分拆 $S^{(r)} = A_1 \cup A_2 \cup \dots \cup A_k$, 存在某个 $i, 1 \leq i \leq k$, 及 S 的某个 q_i 元子集 S_i , 使得 S_i 的所有 r 元子集都属于 A_i 。这个有深刻内涵的组合学定理是 F. P. Ramsey 在 1930 年发表的。当 $r=1$ 时它就是抽屉原理: 把 S 的 n 个元任意分放进编号为 $1, 2, \dots, k$ 的 k 个抽屉, 则当 $n \geq q_1 + \dots + q_k + k - 1$ 时, 必在某个编号为 i 的抽屉中有至少 q_i 个元。当 $r > 1$ 时它是一个典型的“存在性”定理。如记定理肯定其存在的数 N 的最小值为 $R(q_1, \dots, q_k; r)$, 称为拉姆齐数, 虽经长期研究还只确定了极少几个拉姆齐数, 例如至今尚未求得 $R(5, 5; 2)$ 的值。另一方面, 在 1930 年前后, 不同国家的数学家在不同的数学领域独立地发现了几个形色各异但却与拉姆齐定理有内在共性的定理, 它们构成了现称为拉姆齐理论的组合学分支的基础。

组合最优化 根据明确的标准寻求最优或近乎

最优组态的理论与算法,也称组合规划。其研究范围包括大量现实问题,如排序、工作安排、计划制定、装车、选址及路线设计等网络或图上的各种极值问题,这些问题在理论上还可以抽象概括为拟阵上的组合最优化问题。近二三十年来组合最优化是数学规划与计算复杂性理论的交叉领域。

组合几何 组合学和几何学的交叉学科,主要研究以特定类型的几何图形为对象的组态中的各种极值问题。此类问题的最早例子之一是:同时与一个实心球接触且与该球同样大小的实心球最多有几个? J. Kepler 和牛顿都认为是 12 个,但直到 20 世纪中期才严格证明了这一结论。“组合几何”这一术语大约在 20 世纪 50 年代正式出现,并逐渐发展成一个数学分支。

参考文献

1. 柯召,魏万迪. 组合论(上册). 北京:科学出版社,1981
2. 魏万迪. 组合论(下册). 北京:科学出版社,1987
3. Papadimitriou C H, Steiglitz K 著. 组合最优化:算法和复杂性. 刘振宏,蔡茂诚译. 北京:清华大学出版社,1988 (李乔)

zuida gongyinzi

最大公因子 (great common divisor) 能整除两个给定正整数的最大正整数。设 a, b 和 c 都是正整数,若 c 是 a 的因数,又是 b 的因数,则 c 称为 a 和 b 的公因子。如果对于 a, b 的任何公因子 d ,都有 d 整除 c ,则称 c 是 a 和 b 的最大公因子,记作 (a, b) 。 a 和 b 的公倍数中的最小者称为 a 和 b 的最小公倍数,记作 $[a, b]$,显然有 $ab = (a, b)[a, b]$ 。 (a, b) 是 a, b 的所有公因子的倍数, $[a, b]$ 是 a, b 的所有公倍数的因子。若 $a = p_1^{\alpha_1} \cdots p_s^{\alpha_s}, b = p_1^{\beta_1} \cdots p_s^{\beta_s}$ 为 a, b 的因子分解式, p_1, \dots, p_s 为不同的素数, α_i 和 β_i ($i = 1, \dots, s$) 都为非负整数。记 $\gamma_i = \min(\alpha_i, \beta_i), \delta_i = \max(\alpha_i, \beta_i)$, 则 $(a, b) = p_1^{\gamma_1} \cdots p_s^{\gamma_s}, [a, b] = p_1^{\delta_1} \cdots p_s^{\delta_s}$ 。

给定两个正整数 a, b , 一定存在惟一的整数 q_1 和 r_1 , 使 $a = q_1 b + r_1$ ($0 \leq r_1 < b$), q_1 称为以 b 除 a 的部分商, r_1 称为余数。继续进行这样的除法, 设 $b = q_2 r_1 + r_2, r_1 = q_3 r_2 + r_3, \dots$ ($b > r_1 > r_2 > r_3 > \dots > 0$), 这一演算进行有限次后结束, 最后得到 $r_k = q_{k+2} r_{k+1}, r_{k+1}$ 就是 a, b 的最大公因子 (a, b) 。上述计算最大公因子的方法称为欧几里得除法, 也称辗转相除法。

利用欧几里得除法, 也可以找到两个整数 x, y , 使 $ax + by = (a, b)$ 。

对于多项式, 也可以类似地定义它们的最大公因子。设 $f(x)$ 和 $g(x)$ 为两个有理系数多项式, 则必存在两个有理系数多项式 $q(x)$ 和 $r(x)$, 使 $f(x) = q(x)g(x) + r(x)$, 且 $r(x) = 0$, 或者当 $r(x) \neq 0$ 时, $r(x)$ 的次数小于 $g(x)$ 的次数。同样可以利用欧几里得除法计算两个多项式的最大公因子, 并将最大公因子表示成这两个多项式的线性组合。对于任意域上的多项式这个方法也成立。(裴定一)

zuida liu

最大流 (maximum flow) 从源到目的地没有违反各边的容量约束且每个顶点的输入都等于输出的具有最大流值的流。为了严格定义最大流问题, 首先必须定义流网络和流。所谓流网络 $G = (V, E)$ 是一个有向图, 其中每条边 $(u, v) \in E$ 有一个非负容量 $c(u, v) \geq 0$ 。若 $(u, v) \notin E$, 则 $c(u, v) = 0$ 。并且它有两个特殊的顶点, 即源 s 和目的地 t 。为方便起见, 假定对任意 $v \in V$, 存在道路 $s \Rightarrow v \Rightarrow t$ 。故该图连通且 $|E| \geq |V| - 1$ 。设 $G = (V, E)$ 是流网络, 容量函数为 c , 源和目的地分别为 s 和 t 。 G 中的流是实值函数 $f: V \times V \rightarrow \mathbf{R}$, 它具有下面三条性质: ①容量约束: 对所有 $u, v \in V$ 有 $f(u, v) \leq c(u, v)$; ②斜对称: 对所有 $u, v \in V$ 有 $f(u, v) = -f(v, u)$; ③流守恒: 对所有 $u \in V - \{s, t\}$ 有 $\sum_{v \in V} f(u, v) = 0$ 。这时, $f(u, v)$ 称为从顶点 u 到 v 的网络流。流 f 的流值定义为: $|f| = \sum_{v \in V} f(s, v)$ 。所谓最大流问题就是给定流网络 G , 源 s 和目的地 t , 希望求出从 s 到 t 的具有最大流值的流。更一般的, 多源多目的地流网络的最大流问题可转换成上面的单源单目标流网络的最大流问题来求解。

由于管道中液体的流动、电网中电流的配送以及通信网络上信息的传输等现实问题均可抽象为求最大流问题, 所以如何求解最大流问题一度成为研究的热点。先后提出了 Ford-Fulkerson 方法、Edmonds-Karp 算法、预流推进算法和提前算法。求解最大流问题的算法有许多应用。特别地, 最大二分匹配问题可转换成最大流问题来求解。

参考文献

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. Introduction to Algorithms. The MIT Press, 1990 (殷建平 陈火旺)

zuiduan lujing wenti

最短路径问题 (shortest path problem) 寻求加权图中两个指定顶点间加权长度最短的路径的问题(参见图论)。许多最优化问题都能化归为寻求某个加权图的最短路径问题,如运输网络的最低运费路线问题和最快运货路线问题,通信网络的最大可靠性信息传递路线问题和最大期望容量信息传递路线问题,设备更新问题,最佳库存效益问题,市场间运输路线优化表的制定问题,等等。因此对寻求加权图的最短路径,引起了人们的极大兴趣与关注。

给定加权图 $G = \langle V, E \rangle$, 权函数 $w: E \rightarrow \mathbf{R}^+ (\mathbf{R}^+$ 表示非负实数集合) 及 $u, v \in V$ 。现在的问题是: G 中有无从 u 到 v 的路径, 若有则求出一条加权长度最短者。迄今为止, 标号法仍然是求解最短路径问题的最有效算法之一, 其具体作法如下:

若 $i, j \in V$, 则令

$$w_{ij} = \begin{cases} w(e), & \text{若 } e \in E \text{ 为从 } i \text{ 到 } j \text{ 的边} \\ +\infty, & \text{若无从 } i \text{ 到 } j \text{ 的边} \end{cases}$$

第1步 起点 u 标号为零且定标, 其他顶点标号为无穷大且不定标。

第2步 对每个未定标顶点 j 给以临时标号 k_j 使

$$k_j = \min \{b_j, b_i + w_{ij} \mid i \in V \text{ 且 } i \text{ 已定标}\}$$

其中 b_j 为 j 的标号, b_i 为 i 的标号。

第3步 找出临时标号中的最小者, 用它代替相应顶点(当有多个未定标顶点的临时标号均取最小值时, 则任选其中之一)的原标号并定标, 然后撤销所有临时标号。

重复进行第2步和第3步, 直到终点 v 被定标或再没有未定标的顶点能被定标为止。用此法既可判断有无从 u 到 v 的路径, 又能获得从 u 到 v 的加权长度最短的路径。且顶点 v 的标号(当 v 已被定标时)即为最短路径的加权长度。(王兵山)

zuiruo qianzhi tiaojian fangfa

最弱前置条件方法 (weakest precondition method) 基于最弱前置条件的一种程序完全正确性证明方法。最弱前置条件指保证一个语句执行正常结束并满足结果断言的最弱前提条件。它是一个谓词公式, 通常用 $wp(S, R)$ 表示, 这里, R 是语句 S 执行后所期望的结果断言(后置断言)。

E. W. Dijkstra 在前后断言的基础上提出了最弱

前置条件的概念, 以及相应的程序设计演算, 使程序设计和程序验证可同时进行。

对于 E. W. Dijkstra 所定义的语言, 语句的语义通过最弱前置断言给出。 $wp(S, R)$ 可通过逆向推理导出。例如: 赋值语句的语义是 $wp(x := e, R) = R[x/e]$, 即将 R 中 x 的所有自由出现同时代换成 e 。例如:

$$wp("x := x * x", x^4 = 10) = ((x * x)^4 = 10) = (x^8 = 10)$$

为了证明循环的终止性, E. W. Dijkstra 引入了循环不变式和界函数。一般说来, 一个循环呈如下形式:

$\{invariant: P\}$ — 进入循环前, 不变式 P 真,
 $\{bound: t\}$ — 并且 B 真时 $t > 0$, t 是循环次数的上界

do $B \rightarrow$ Decrease t , Strue od

— 当 B 真时, 使 t 递减并执行 S , S 执行过程真

保持 P

$\{P \wedge \neg B\}$ — 则循环必然终止且终止时 P 真 B 假

若 Q 是 S 的执行能在有限时间内中止并满足 R 的任一前提条件, 则必有 $Q \Rightarrow wp(S, R)$ 。因此, 证明前后断言 $Q \{S\} R$ 只需先求出最弱前置断言 $wp(S, R)$, 再证明 $Q \Rightarrow wp(S, R)$ 。

当给定了 Q 和 R , 根据 Q, R 的结构, 通过推导 $wp(S, R)$, 可推出 S 的结构, 从而将程序设计的过程变成数学推导的过程。例如, 要设计一个循环 DO, 使得当满足前置断言 Q 和结果断言 R , 则 P, t 和 B 应满足 $Q \Rightarrow P \wedge bound: t, t \leq 0 \Rightarrow \neg B$ 及 $P \wedge \neg B \Rightarrow R$ 。这实际上给出了循环语句设计的原则。

参考文献

1. Dijkstra E W. A Discipline of Programming. Englewood Cliffs: Prentice Hall, 1976
2. Gries D. The Science of Programming. New York: Springer - Verlag, 1981 (伊波)

zuixiao erchengfa

最小二乘法 (method of least squares) 用有限个线性无关函数的线性组合所做的最佳平方逼近。

根据给定的一组数据: $(x_i, y_i), i = 1, 2, \dots, m$, 在某特定的函数类中找一函数 $f(x)$, 使偏差平方和

$\sum_{i=1}^m [f(x_i) - y_i]^2$ 达到极小。有时,由于各点数据可靠性不一致,故可引进权数 $d_i > 0, i = 1, 2, \dots, m$,

从而函数 $f(x)$ 可由求带权偏差平方和 $\sum_{i=1}^m d_i [f(x_i) - y_i]^2$ 达到极小而得出。这种找 $f(x)$ 的方法称为最小二乘法。它是 19 世纪初由 A. M. Legendre 和 C. F. Gauss 分别独立提出的。科学实验与工程技术中的许多经验公式大都应用此方法建立。一般取

$f(x) = \sum_{j=1}^n \alpha_j \varphi_j(x)$, 其中 $\alpha_1, \alpha_2, \dots, \alpha_n$ 为待定参数; $\varphi_1(x), \varphi_2(x), \dots, \varphi_n(x)$ 为选定的一组线性无关的函数,称做基函数,且 $m \geq n$ 。令

$$c_{ij} = \varphi_j(x_i) \quad (i = 1, 2, \dots, m; j = 1, 2, \dots, n)$$

记

$$C = \{c_{ij}\} \in R^{m \times n}, \quad \alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)^T,$$

$$y = (y_1, y_2, \dots, y_m)^T$$

则最小二乘法可表示为:求向量 α , 使

$$\|C\alpha - y\|_2^2 \quad (1)$$

达极小,由微分学极小值的方法可推得 α 满足方程组

$$C^T C \alpha = C^T y \quad (2)$$

此方程称为最小二乘法方程组,它的解为最小二乘解。当 C 为列满秩时,方程组 (2) 有惟一解,若 C 不是列满秩,则 (2) 的解不惟一,但具有最小欧氏长度的解是惟一的。该解称为极小最小二乘解。不论何种情况,最小二乘问题 (1) 的解可表示为

$$\alpha = C^+ y$$

其中 C^+ 为 C 的广义逆矩阵(在 Moore-Penrose 意义下)。但 m 较大时, $C^T C$ 往往是病态矩阵,因而直接求解 (2) 是不利的,此时,可对 C 进行 QR 分解,即存在 $m \times n$ 阶正交矩阵 Q 及 $n \times n$ 阶上三角矩阵 R , 使

$$Q^T C = \begin{pmatrix} R \\ 0 \end{pmatrix}$$

则方程

$$R\alpha = y_1^*$$

的解就是所求的最小二乘解,其中 y_1^* 由向量 $y^* = Q^T y$ 的前 n 个分量组成。

基函数 $\varphi_j(x)$ 的选取是极为重要的,最常用的方法如下。

多项式最小二乘法 取基函数 $\varphi_j(x) = x^{j-1}$, $j = 1, 2, \dots, n$

$$f(x) = \sum_{j=1}^n \alpha_j x^{j-1}$$

但当 $n \geq 6$ 时的相应法,方程是病态的,计算时可取 $\varphi_j(x)$ 为正交多项式,或通过给定点列 $\{x_i\}_{i=1}^m$ 构造按点列正交的多项式。

样条最小二乘法 当数据 $(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$ 较多且有较多波动起伏时, $f(x)$ 可假设为分段多项式,特别是三次样条 $S_3(t_0, t_1, \dots, t_r)$, 其中

$$\dots < t_{-2} < t_{-1} < t_0 < t_1 < \dots < t_r < t_{r+1} < \dots$$

且使 x_1, x_2, \dots, x_m 都落在 $[t_0, t_r]$ 之上。令

$$\psi_{3,i}(x) = \frac{1}{2} \sum_{j=i}^{i+4} \beta_{i,j} |x - t_j|^3,$$

$$t = -3, -2, \dots, r-1$$

$$\beta_{i,j} = \frac{4!}{\omega'_i(t_j)}, \quad \omega'_i(t_j) = \prod_{\substack{k=i \\ k \neq j}}^{i+4} (t_j - t_k)$$

取基函数 $\varphi_j(x) = \psi_{3,j-4}(x)$, $j = 1, 2, \dots, n$

则得到 $f(x) = \sum_{j=1}^n \alpha_j \varphi_j(x) = \sum_{j=1}^n \alpha_j \psi_{3,j-4}(x)$,

$$t_0 \leq x \leq t_r = t_{n-3}$$

参考文献

1. Lawson C L and Hanson R J. Solving Least Squares Problems. Englewood Cliffs, New Jersey: Prentice-Hall, 1974

2. 冯康等. 数值计算方法. 北京: 国防工业出版社, 1978 (沈毅)

zuixiao shengcheng shu

最小生成树(minimum spanning tree) 由给定图的所有顶点和部分边形成的总权最小的树。具体来说,给定连通的无向图 $G = (V, E)$, 其中 V 为顶点集, E 为边集, 并给定权函数 $W: V \times V \rightarrow R$, 其中 R 为实数集, 图 G 的最小生成树就是 E 的无圈子集 T 使得 T 连接所有顶点且其总权 $W(T) = \sum_{(u,v) \in T} W(u,v)$ 最小。显然,一个图的最小生成树可能不惟一。

由于输电线路的架设、公路体系的构建以及计算机网的连接等现实问题均可抽象为求最小生成树的问题,所以如何形成一棵最小生成树的问题一度成为研究的热点。一种一般的求解思路是:将集合 A 初始化为空集 \emptyset , 当 A 不是生成树时,求出对 A 安全的边 (u,v) 并把它加入 A 中,直到 A 是生成树为止。这里,边 (u,v) 对集合 A 安全是指 $A \cup \{(u,v)\}$ 仍是某棵最小生成树的子集。因此,求出对 A 安全

的边 (u, v) 成为求解最小生成树问题的关键。可以证明,如果 $G = (V, E)$ 是连通的无向图,且有定义在 E 上的实值权函数 W , A 是 E 的子集并包含在 G 的某棵最小生成树中, $(S, V-S)$ 是重视 A 的 G 的任何割, (u, v) 是横过 $(S, V-S)$ 的轻边,那么边 (u, v) 对 A 是安全的。这里,无向图 $G = (V, E)$ 的“割” $(S, V-S)$ 是指 V 的一个划分。边 $(u, v) \in E$ 横过割 $(S, V-S)$ 是指定的一个端点在 S 中,而另一个端点在 $V-S$ 中。一个割重视边集 A 是指没有 A 中的边横过这个割。一条边是横过某个割的轻边是指其权是横过该割的所有边中的最小者。显然,轻边可能不惟一。根据上述结果求解时,每次挑选的安全边是轻边,所以形成的求解算法便是贪心算法。执行过程中,集合 A 总是无圈的。图 $G_A = (V, A)$ 形成一个森林。任意对 A 安全的边 (u, v) 连接 G_A 的不同连通分支。开始时 $A = \emptyset$, G_A 中有 $|V|$ 棵树,每次挑选一条边,森林中树的数目减1,挑选出 $|V| - 1$ 条边后,整个森林只含一棵树,这棵树就是要求的最小生成树,算法终止。从上述结果还可看出,定义不同的割,便可求出不同的轻边,从而得到不同的求解算法。历史上最有名的是克鲁斯卡尔算法和普列姆算法。在克鲁斯卡尔算法中,集合 A 是一个森林,加到 A 中的安全边总是连接两个不同分支的图的最小权边。而在普列姆算法中,集合 A 形成一棵树,加到 A 中的安全边总是连接边树与不在这棵树上的某个顶点的最小权边。

参考文献

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. Introduction to Algorithms. The MIT Press, 1990
(殷建平 陈火旺)

zuiyouhua fangfa

最优化方法 (optimization method) 寻求(逼近)某函数(目标函数)在某些附加条件(约束条件)下的最优(极大或极小)值的计算方法。最优化的两个主要分支是动态最优化(与时间有关)与静态最优化,后者通常称为数学规划。1939年П. Б. Канторович首先研究和应用了线性规划方法,1947年G. B. Dantzig提出了单纯形方法,他们为线性规划的最优化方法奠定了基础。1951年H. W. Kuhn和A. W. Tucker给出了非线性规划的基本定理,为数学规划奠定了理论基础。随着计算机技术的飞速发展,最优化方法已在工程、军事、生产、管理和经济等方面得到了广泛的应用。运筹学中所处理的问题

绝大部分是数学最优化问题。用来解决数学最优化问题的方法,例如数学规划(包括线性规划和非线性规划)、排队论与决策分析等都属于最优化方法范畴。最优化方法还涉及工程控制、最优控制、系统科学等。

无约束最优化方法 寻求(逼近)多元函数 $f(\mathbf{x})$ 在整个实 n 维空间 R^n 中的局部极小值点的计算方法称为**无约束最优化方法**。它也是许多约束最优化方法的基础。大多数无约束最优化方法都是迭代法,每一次迭代都从某一点 $\mathbf{x}^{(k)}$ 移到另一个适合条件 $f(\mathbf{x}^{(k+1)}) < f(\mathbf{x}^{(k)})$ 的点 $\mathbf{x}^{(k+1)}$ 。为了得到 $\mathbf{x}^{(k+1)}$,首先要确定移动的方向 $\mathbf{S}^{(k)}$,其次要确定沿方向 $\mathbf{S}^{(k)}$ 移动的步长 λ_k ,于是 $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \lambda_k \mathbf{S}^{(k)}$ 。对应于 $\mathbf{S}^{(k)}$ 与 λ_k 的不同选取,就得到不同的算法。算法可分为两大类:一类是直接法,一般对目标函数的解析性质不作苛刻要求,常用于变量不多、函数比较复杂或不易计算偏导数的情形。较为常用的直接法有鲍威尔法、单纯形法、模式搜索法和罗森布罗克法;另一类是解析法,要用到目标函数的(偏)导数。

最速下降法和牛顿法 在要求计算(偏)导数值的算法类中,一般取移动方向 $\mathbf{S}^{(k)}$ 满足

$$(\mathbf{S}^{(k)})^T \nabla f(\mathbf{x}^{(k)}) < 0$$

这里 T 表示矩阵(或向量)的转置运算, $\nabla f(\mathbf{x})$ 表示 $f(\mathbf{x})$ 在点 \mathbf{x} 上的梯度,即

$$\nabla f(\mathbf{x}) = (\partial f(\mathbf{x}) / \partial x_1, \partial f(\mathbf{x}) / \partial x_2, \dots, \partial f(\mathbf{x}) / \partial x_n)^T$$

最速下降法采取如下的迭代步骤:首先选取初始点 $\mathbf{x}^{(0)} \in R^n$ 及判别收敛的正数 ε ,记 $\mathbf{S}^{(0)} = -\nabla f(\mathbf{x}^{(0)})$ 。其次,当 $\mathbf{x}^{(k)}$ 已知,且 $\mathbf{S}^{(k)} = -\nabla f(\mathbf{x}^{(k)})$,若 $\|\mathbf{S}^{(k)}\| < \varepsilon$,则取 $\mathbf{x}^{(k)}$ 为近似解;若 $\|\mathbf{S}^{(k)}\| \geq \varepsilon$,则采用单变量函数求极值的方法,并称之为线性搜索求 λ_k ,使

$$\min_{\lambda \geq 0} f(\mathbf{x}^{(k)} + \lambda \mathbf{S}^{(k)}) = f(\mathbf{x}^{(k)} + \lambda_k \mathbf{S}^{(k)})$$

令 $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \lambda_k \mathbf{S}^{(k)}$,重复这一步骤,直到求得满足误差范围的近似最优解为止。在该方法中,取移动方向为 $\mathbf{S}^{(k)} = -[\nabla^2 f(\mathbf{x}^{(k)})]^{-1} \nabla f(\mathbf{x}^{(k)})$ 所构成的迭代方法称之为牛顿法。这里 $\nabla^2 f(\mathbf{x}^{(k)})$ 为 $f(\mathbf{x}^{(k)})$ 的二阶偏导数矩阵。

共轭梯度法和变尺度方法 该法既有较快的收敛速度又无需计算二阶偏导数。共轭梯度法是由R. Fletcher和C. M. Reeves于1964年提出的,首先从任意的初始点 $\mathbf{x}^{(1)}$ 开始,在第 k 次迭代时取移动

方向

$$S^{(k)} = \begin{cases} -\nabla f(\mathbf{x}^{(k)}) & (k=1) \\ -\nabla f(\mathbf{x}^{(k)}) + \frac{\nabla f(\mathbf{x}^{(k)})^T \nabla f(\mathbf{x}^{(k)})}{\nabla f(\mathbf{x}^{(k-1)})^T \nabla f(\mathbf{x}^{(k-1)})} S^{(k-1)} & (k>1) \end{cases}$$

然后再作线性搜索得到 λ_k 和 $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \lambda_k S^{(k)}$ 。对于二次凸函数只要有限步就能达到最小值点 \mathbf{x}^* ；对于非二次函数，常采用所谓周期性重开始的策略，这在理论上可以证明达到 n 步二阶收敛，若不采取这一策略，其收敛阶为线性的。变尺度方法首先由 W. C. Davidon 于 1959 年提出，后来相继形成两种常用的变尺度方法，分别称之为 DFP 算法和 BFGS 算法，它们的共同特点是：移动方向由 $S^{(k)} = -H_k \nabla f(\mathbf{x}^{(k)})$ 确定，其中 H_k 为 n 阶对称正定方阵； H_1 可以任意选取，但通常取 $H_1 = I$ 。这两个算法只在 $H_k (k>1)$ 的定义方法上有所不同。可以证明在一定的条件下，它们都是超线性收敛，且都是 n 步二阶收敛，后者与前者相比有较好的数值稳定性。

约束最优化方法 寻求(逼近)目标函数 $f(\mathbf{x})$ 在约束条件 $g_i(\mathbf{x}) \leq 0 (i=1, 2, \dots, m)$ 下的最优(极小)值点问题的计算方法称为约束最优化方法。这里 $f(\mathbf{x})$ 与 $g_i(\mathbf{x})$ 都是 R^n 中的实值函数，当 $f(\mathbf{x})$ 和 $g_i(\mathbf{x})$ 均为线性函数时，上述问题称为线性规划，否则称为非线性规划。非线性规划中的一个分支是凸规划，它的目标函数 $f(\mathbf{x})$ 与约束函数 $g_i(\mathbf{x})$ 都是凸函数。介于线性规划与凸规划之间的是二次规划，它的目标函数是正定二次型，约束是线性的。若对上述问题附加条件——某些变量或全部变量只能取整数值——则称为整数规划。此外还有随机规划、几何规划和多目标规划等。

求解线性规划问题的最常用算法是单纯形法。1963 年 P. Wolfe 将单纯形方法作了修改，用以求解二次规划，该法只经过有限次迭代即可求得最优解。1984 年 N. Karmarkar 提出一种关于线性规划的多项式时间的算法，这在理论上和应用上都是很有价值的。

可行方向法 该法根据逐次沿可行方向求可行解点的迭代思想构造一点列 $\{\mathbf{x}^{(k)}\}$ ，使其满足某种给定要求。其关键在于适当选取向量 $S^{(k)}$ ，使点列 $\{\mathbf{x}^{(k)}\}$ 符合一般迭代法的要求。对线性约束的情形，当 $f(\mathbf{x})$ 为可微凸函数时有三种较为有效的求 $S^{(k)}$ 的算法：①由 G. Zoutendijk 于 1960 年提出的可行方向法，取 $S^{(k)} = \mathbf{y}^{(k)} - \mathbf{x}^{(k)}$ ，其中 $\mathbf{y}^{(k)}$ 为 $f(\mathbf{x})$ 在

$\mathbf{x}^{(k)}$ 处的线性近似函数 $f(\mathbf{x}^{(k)}) + \nabla f(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)})$ ，在线性约束下达到最小值的最优解。②由 J. B. Rosen 于 1960 年提出的梯度投影法，运用在 $\mathbf{x}^{(k)}$ 处投影矩阵 P_k 的公式，取 $-P_k \nabla f(\mathbf{x}^{(k)})$ 为 $S^{(k)}$ ，从而避免了每迭代一次就要解一个线性规划的手续。③由 P. Wolfe 于 1963 年提出的既约梯度法，它应用消去基变量的方法和 $f(\mathbf{x})$ 的既约梯度的概念构造出 $S^{(k)}$ 。这三种方法所产生的点列 $\{\mathbf{x}^{(k)}\}$ 虽然可以使函数值序列 $\{f(\mathbf{x}^{(k)})\}$ 单调下降，但却不一定收敛于最优解。随后陆续有不少研究工作，对原有方法进行种种修正，从而得出具有收敛性的各种新方法。1969 年 D. Goldfarb 结合梯度投影法与变尺度法提出了一种可行方向法，对二次凸规划是有限步收敛的，而且可以推广处理非线性约束的情形。同年，由 J. Abadie 和 J. Carpentier 将既约梯度法加以推广，提出了广义既约梯度 (GRG) 法，用来求解具有非线性约束的最优化问题，它是解一般非线性规划问题的一种很好的算法。

上述线性近似型算法的收敛速度，一般都不高于超线性的。对于二阶可微的函数 $f(\mathbf{x})$ ，在 $\mathbf{x}^{(k)}$ 处若用二次函数 $f(\mathbf{x}^{(k)}) + \nabla f(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)}) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^{(k)})^T \nabla^2 f(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)})$ 来近似，进而对可微函数 $f(\mathbf{x})$ 用种种变尺度矩阵 H_k 去代替近似式中的二阶偏导数矩阵 $\nabla^2 f(\mathbf{x}^{(k)})$ ，将约束问题的求解化为求一系列二次规划的解，这类方法统称为序贯二次规划法。利用计算过程中得到的信息和变尺度公式来更新 H_k ，这种逐次二次规划算法也称为约束变尺度算法，它是求解带非线性约束的最优化问题的重要方法之一^[1]。

序贯无约束极小化方法 该法将求解约束极值问题转化为一系解无约束的极值问题。对于具有不等式约束 $g_i(\mathbf{x}) \leq 0 (i=1, 2, \dots, m)$ 的非线性规划问题，作函数 $p(\mathbf{x}, t) = f(\mathbf{x}) + t \sum_{i=1}^m [\max(g_i(\mathbf{x}), 0)]^2$ ，在适当的假设下， $p(\mathbf{x}, t)$ 是对 \mathbf{x} 不加约束下的最优解 $\mathbf{x}(t)$ ，在 $t \rightarrow \infty$ 时趋于原问题的最优解。这种方法称为罚函数法。1969 年 M. R. Hestenes 和 M. J. D. Powell 结合拉格朗日乘子法和罚函数法的特点，对约束为等式的情形，提出了可微的增广拉格朗日函数，并指出在适当的假设下，能够求得原问题的最优解。此后陆续有不少工作对一般可微非线性规划构造了各种不同的可微增广拉格朗日函数，并给出了算法的迭代程序，这类方法统称为广义乘子

法。鉴于罚函数法产生的点列 $\{x(t_k)\}$ 是从约束集的外部来逼近约束边界上的最优解,提出所谓障碍函数 $B(x, r) = f(x) - r \sum_{i=1}^m \ln(-g_i(x))$ ($r > 0$), 可使 $B(x, r)$ 的无约束最优解 $x(r)$ 在约束集内达到, 当 $r \rightarrow 0$ 时, $x(r)$ 趋于原问题的最优解。这种方法称为障碍函数法。另外还有一种典型的障碍函数, 其表达式为

$$B(x, r) = f(x) - r \sum_{i=1}^m (1/g_i(x)).$$

对兼有等式和不等式约束的最优化问题, 可结合使用罚函数与障碍函数而构造出混合型函数来求解, 1969 年 W. I. Zangwill 提出用统一的观点研究算法, 他的基本思想是将算法视为一个点列集的映像。在一些假设下由上半连续的点到集映像产生的点列 $\{x^{(k)}\}$ 收敛于最优解。从而统一了不少已有算法的收敛性的研究。这方面的工作还在不断发展。

参考文献

1. Fletcher R. Practical Methods of Optimization. Unconstrained Optimization, 1979, 1, New York: John Wiley & Sons

2. Avriel M 著. 非线性规划——分析与方法. 李元嘉等译. 上海: 上海科学技术出版社, 1979

(汪裕武)

zuoye

作业 (job) 用户请求计算机系统完成的一个计算任务。它由用户程序、数据及其所需的控制命令组成, 每个作业一般可分成若干顺序处理和加工的步骤, 称为作业步。例如, 某个用 FORTRAN 语言编写的计算问题, 可经过以下作业步来求解: 编辑、编译、连结、装入、运行。

在批处理环境下, 一批作业有序地排列起来, 分批输入计算机, 再由作业调度选择并启动运行, 形成了一个自动处理作业的**作业流**。用户根据系统提供的手段, 对其作业在系统中的整个运行过程所进行的控制叫**作业控制**。作业控制由作业控制语言 (JCL) 来描述。JCL 由作业控制语句组成, 每个 JCL 语句是一行传送作业步信息的编码, 可穿在控制卡上, 也可通过键盘以命令方式发送。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989

(郑宇华)

zuoye guanli chengxu

作业管理程序 (job manager) 操作系统中负责所有作业的从提交到完成期间的组织、管理和调度工作的程序。

作业管理具有两类功能: 作业调度和作业控制。按照一定调度策略, 从输入井的后备作业队列中选择作业进入主存运行的工作称**作业调度**。它完成作业从收容状态到执行状态、从执行状态到完成状态的转变。具体地说: 按预定算法, 从收容状态的后备作业中选择作业投入运行; 记录进入系统的作业的情况, 并为每个作业建立作业控制块; 为选中作业分配所需资源, 为该作业建立用户进程; 做好作业运行结束的善后处理。

选择和确定作业调度算法十分复杂, 可以用以下指标来评估它的好坏: 平均作业周转时间、吞吐量和中央处理器 (CPU) 利用率。

常用的作业调度算法有: 先来先服务, 最短作业执行时间优先, 响应比最高优先, 优先数法, 分类调度法。

作业控制指用户使用操作系统提供的手段和设施来组织和控制用户作业的运行, 即用户对作业上机操作的全过程的各种干预。例如, 如何组织作业? 如何输入作业? 如何执行作业? 如何处理异常? 等等。可以分成两类作业控制方式:

(1) **脱机作业控制** 又称**批处理**。在这种方式下, 用户应使用作业控制语言的语句, 把要求计算机系统执行的工作穿成控制卡或写成作业说明书, 连同程序、数据一起提交给计算机。当作业调度选择作业并投入运行后, 作业控制程序逐条解释并执行作业控制语句, 以实现**对作业的控制**。作业运行过程中, 用户不得对作业进行干预, 故称**脱机控制**。这种方式的优点是: CPU 和资源利用率、作业吞吐量大。批处理操作系统均采用这种方式。

(2) **联机作业控制** 又叫**交互型处理**。在这种方式下, 允许多个联机用户通过终端共享一台计算机系统, 用户在联机终端上使用系统提供的终端操作命令, 逐条打入, 逐条执行, 系统及时将执行情况和运行结果反映给用户, 形成了一个**人机组成的闭合系统**, 能充分发挥人的主观能动性, 可动态、交互地控制作业运行。广泛应用于分时系统、个人计算机系统和工作站。

联机作业控制可分成以下三种: ①**键盘命令方式**, 用户通过键盘向计算机系统发出各种命令, 系统执行用户的各种要求。②**命令文件方式**, 用户按作

业上机处理的控制步骤,把需键入的操作命令,事先编好输入并保存到一个文件中。这个文件称命令文件。其中,每一行都是一条键盘命令。执行命令文件时,只要按格式打入命令文件的文件名和附加参数,系统便可自动读取命令文件内容,并逐条解释执行,直至结束。③选单驱动方式,这种方式以层次式逐级选单为引导,用户不必键入命令,只需选择选单项便可执行各种交互式任务,给用户使用计算机系

统带来极大方便,特别适合经验不足的程序员或非计算机专业人员使用。

随着多媒体技术的发展和成熟,趋向于利用声音、文字、图形、图像等形式进行数据输入输出和联机作业控制。

参考文献

孙钟秀等. 操作系统教程. 北京: 高等教育出版社, 1989
(费翔林)

外文字头

A* suanfa

A* 算法 (A* algorithm) 能够对于存在解的状态搜索问题,较快地找到一条最佳求解路径,即最小费用解径的启发式搜索算法,是一般有向图启发式搜索——A 算法的改进算法。

在人工智能领域进行问题求解时,通常可以把数据库的状态看作一个节点,把数据库所有可能状态的全体看作一个隐含的问题空间,并表示为一个有向图。图中每条弧代表对数据库的一个操作,它可以使一个状态转换为另一个状态。如果从代表初始状态的起始节点出发,有一条路径通向目标状态节点,则称此目标状态所代表的问题在给定的初始状态下有解,而从初始状态到目标状态通路上的每一条通路,即操作符序列,就构成了问题的一条解径,或解过程。

对于存在有解的人工智能问题,虽然用盲目搜索方法,如宽度优先搜索法等,也能得到问题的最佳解径,但是由于在搜索过程中需要展开过多的节点,因而限制了可求解问题的规模。另一方面,若一般性地利用与问题有关的启发信息进行搜索,虽可大大地提高搜索效率,但却不能保证一定能够找到最小费用解径。A* 算法就是要综合考虑路径本身的计算费用以及寻求路径所需的费用这两方面的因素,以适度增加搜索工作量为代价,来保证找到问题的一条最佳解径。

在启发式搜索中,需要有一种计算从各待扩展节点出发,达到某种预定指标的期望程度的方法,以便能优先扩展那些最有希望的节点。一种常用的方法就是定义称为估价函数的实值函数 f 。

A 算法及 A* 算法定义的估价函数 $f(n)$ 如下:

设函数 $f^*(n)$ 是从初始节点 S 出发约束通过给定节点 n ,然后到达目标节点 t 的实际最小耗费路径的费用, $g^*(n)$ 是从初始节点 S 到节点 n 的最小耗费路径的费用, $h^*(n)$ 是从节点 n 到目标节点 t 的最小耗费路径的费用,并且 $f^*(n) = g^*(n) + h^*(n)$,即节点 n 在最佳路径上的实际费用 $f^*(n)$ 等于从初始节点 S 到节点 n 的最佳路径的费用与从节点 n 到目标节点 t 的最佳路径的费用之和。那么

$f^*(n)$ 的估价函数 $f(n)$ 为

$$f(n) = g(n) + h(n)$$

其中, $g(n)$ 是迄今为止已找到的从初始节点 S 到节点 n 的最小耗费路径的费用,它被视为 $g^*(n)$ 的一个估计,满足 $g(n) \geq g^*(n)$; $h(n)$ 是 $h^*(n)$ 的一个估计,其定义取决于与问题有关的某些信息,例如,从节点 n 到达目标节点 t 的距离;节点 n 与目标节点 t 的差异等。 $h(n)$ 常被称为启发式函数。

在图搜索过程中, A 及 A* 算法利用节点的估价函数值 $f(n) = g(n) + h(n)$ 对 OPEN 表中的待扩展节点进行排序。节点 n 的估价函数值 $f(n)$ 越小,它处于最佳路径的可能性越大,越应排在序列的前面,优先扩展。但与 A 算法不同的是, A* 算法还进一步要求所定义的启发式函数 $h(n)$ 为函数 $h^*(n)$ 的下界,亦即对图中所有节点 n , 均有 $h(n) \leq h^*(n)$ 。这样,如果给定问题确有解存在,那么 A* 算法就可如此通过减弱启发式信息量,适当地降低搜索效率,确保找到问题的最佳解路径。例如,在极端情况下,即 $h(n) = 0$ (肯定满足下界条件), $g(n) = d(n)$ (节点 n 的深度) 时, A* 算法等同于宽度优先算法,而宽度优先算法总能找到达到目标节点的最小长度路径。

下面,我们给出 A* 算法的一般执行过程。

(1) 建立搜索图 G 和 OPEN 表 (开始它们仅含初始节点 S), 并计算 S 节点的估价函数值 $f(S)$;

(2) 建立 CLOSED 表 (开始时为空表);

(3) 若 OPEN 表为空表,无解,失败退出;

(4) 取 OPEN 表中第一个元素为当前扩展节点 n , 并将其放入 CLOSED 表中;

(5) 若 n 为目标节点,有解,成功返回。其解为图 G 中从节点 S 到 n 沿指针所得的路径 (指针由第 (7) 步确定);

(6) 扩展节点 n , 生成不是 n 的祖先的所有后继节点 $\{m_i\}$, 计算各后继节点的估价函数值 $f(n, m_i) = g(n, m_i) + h(m_i)$, 其中, $g(n, m_i)$ 是初始节点 S 通过 n 到 m_i 的费用, $f(n, m_i)$ 是 S 通过 n, m_i 到目标节点费用的估计。将它们作为节点 n 的后继节点添入图 G 中;

(7) 标记或修改后继节点集合 $\{m_i\} = \{m_j\} \cup \{m_k\} \cup \{m_l\}$ 中各节点的指针。这里, m_k 为已在 OPEN 表中出现过的待扩展节点, m_l 为已在 CLOSED 表中出现过的已扩展节点, m_j 为从未在二表中出现过的第一次生成的节点;

对于所有节点 $m_j \in \{m_j\}$, 建立 m_j 到节点 n 的指针, 并将它们添入到 OPEN 表中;

对于所有节点 $m_k \in \{m_k\}$, 比较约束通过 n 的 m_k 节点的估价值 $f(n, m_k)$ 与在扩展 n 之前已计算出的 m_k 的估价值 $f(m_k)$ (假设还有其他从 S 到 n 的通路), 若 $f(n, m_k)$ 较小时, 则令 $f(n, m_k)$ 为节点 m_k 的估价函数值, 即 $f(m_k) = f(n, m_k)$, 并修改 m_k 的指针, 使其指向 n ;

对于所有节点 $m_l \in \{m_l\}$, 操作同 m_k 节点。并且, 当 $f(n, m_l)$ 值较小时, 还要把 m_l 节点放回到 OPEN 表中, 以便能重新扩展 m_l 以及计算并修改其后继节点的估价函数值和指针;

(8) 将 OPEN 表中节点按 f 值从小到大顺序重新排列;

(9) 转向(3)。

为了提高 A* 算法的执行效率, 人们已经证明, 如果启发式函数 h 能进一步满足单调限制条件, 亦即, 如果对于所有节点 n_i 和 n_j (n_j 是 n_i 的一个后继节点) 总有

$$h(n_i) - h(n_j) \leq C(n_i, n_j)$$

并且

$$h(t) = 0$$

式中 $C(n_i, n_j)$ 为 n_i 到 n_j 的弧费用;

t 为目标节点。

那么, 每当 A* 算法选择某个节点进行扩展时, 也就是它已找到了从初始节点 S 到达该节点的最佳路径。这样就没有必要反复地扩展 CLOSED 表中的节点, 因而可以删除上面算法步骤(7)中的最后一步了。

参考文献

1. Nilsson N J 著. 人工智能原理. 石纯一等译. 北京: 科学出版社, 1984
2. 林尧瑞, 马少平. 人工智能导论. 北京: 清华大学出版社, 1989 (杨莉 康建初)

Ada yuyan

Ada 语言 (Ada language) 一种通用程序设计语言。它具有如 PASCAL 等通用语言和某些专用语言的长处, 有通用控制结构, 有定义数据类型和分

程序的能力, 且易于控制并行任务和处理异常情况。所以它可用于数值分析计算, 系统程序设计, 并满足实时和并行操作等要求。Ada 语言既适于军用, 也适于民用。

在 20 世纪 70 年代初, 美国国防部为摆脱软件费用急剧增长的困境, 提出设计研制统一的军用结构语言。其需求包括: 可靠性、易维护性、结构化程序的构造、强数据类型、相对精度和绝对精度的规约、信息隐蔽和数据抽象、并发处理、异常处理、类属(式样)定义、与机器有关的设施等。美国军方认为没有任何现存的语言能满足所有这些需求, 所以招标设计。1979 年 4 月最后选定由法国 Jean Ichbiah 教授领导的设计小组所设计的“绿色语言”, 并取名为 Ada 语言, 以纪念世界上第一位有文字记载的女程序员 Augusta Ada Lovelace (1815 年—1852 年)。

1980 年 7 月出版了 Ada 语言手册, 年末提供了编译系统。1983 年 Ada 语言被正式列入美国标准 (ANX1/MIL 1815A—1983), 后来先后被批准为美国联邦标准和国际标准 (ISO/IEC 8652: 1987—1992)。美国国防部规定只有经测试合格者方可被命名 Ada 编译。从 1991 年起, 许多国家的军方规定使用 Ada 语言作为惟一计算机程序设计语言。Ada 产品和应用不断增加。由于使用过程中的反馈意见, 计算机硬件的飞速发展和软件技术的进步, 如面向对象技术等, 促使把 Ada 83 修改成 Ada 95, 其内容要点如下:

(1) 扩展程序设计 允许继承、修改或增加父类型的已有分量和运算, 目的是可以重复使用现有可靠的软件, 不必重新编译和测试。

(2) 宽类程序设计 引入宽类类型以能处理同一类中的任何一种类型。至今我们所接触到的功能允许定义新类型为现有类型的扩展, 例如我们引入了几种警戒类型, 它们各不相同又相关联。现在引入宽类类型, 就能操作任何一种警戒类型并相应地处理。

(3) 抽象类型和子程序 抽象类型的目的是为通过派生进一步建立有用的类型提供共同的基础。抽象子程序是一种占位符号, 可以提供运算(它没有体)。

(4) 类型扩展 引入加标记类型的概念。以上都是有关类型扩展的, 好处是不必重新编译、测试现有稳定的系统, 可重复使用, 这是面向对象语言最重要的特性。

(5) 动态选择 一种灵活的访问子程序机制, 可

以把子程序作为参数来传递,动态绑定(binding)。

(6) 其他访问类型 除了可访问子程序外,还可访问对象。

(7) 层次库 Ada 95 引入层次库,包含子代包和子程序。这样逻辑上不同的包可共享一个私有类型扩展包而不必重新编译。子代又可以有子代,带来许多灵活和方便。子代分公有、私有两种。

(8) 私有子代单元 私有子代的说明对用户命令是可见的,私有子代就不给用户任何附加的可见性,这便于把大程序分层。

(9) 保护类型 封装并提供对类型的永有对象数据同步访问,而不必引入附加的任务。保护类型为多个任务同步存取共享数据提供了有效的手段,这是实时系统的一个关键要求。

(10) 任务调度和计时 Ada 83 的调度规则对于汇合特别不能令人满意,Ada 95 允许更自由地优先度选择和调度规则。

(11) 类属参数 Ada 95 中能声明形式类属包。

(12) 其他改进 引入控制类型,给出初始化、终止以及用户定义的赋值,以及使用访问值判别项给出继承功能。一些关于数组的限制被放松。包含了对无符号整型的支持,提供了移位和逻辑运算。这些都大大地减轻了系统程序员的负担。

(13) 预定义库 由于 Ada 95 提供层次库的概念,所以有许多标准库中提供的附加预定义包已重新构造。

(14) 专门需要的附录 Ada 95 中有 6 个专门需要的附录:

- ① 系统程序设计;
- ② 实时系统;
- ③ 分布式系统;
- ④ 信息系统;
- ⑤ 数值;
- ⑥ 安全性和保密性。

参考文献

1. Ada 95 Reference Manual. International Standard ISO / IEC - 8652; 1995, Jan. 1995
2. Ada 95 Rationale. Intermetrics, Inc. Jan. 1995
3. 袁崇义,徐泽同译. 程序设计语言 Ada 参考手册(Ada 83). 北京:科学出版社,1986
4. 中科院软件所 8 室 Ada 9 × 研究小组译. Ada 9 × 项目报告:映象文件. 北京:计算机研究与发展《增刊》,1993 (程虎)

ALGOL 60 yuyan

ALGOL 60 语言 (ALGOL 60 language)

一种高级程序设计语言。ALGOL 是 algorithmic language 的缩略语。

1958 年在苏黎世举行的一次 ACM 小组和 GAMM 小组的联合会议上(GAMM 为欧洲小组的正式名称),两个小组把他们关于算法表示法的建议综合为一,于是就产生了 ALGOL,又称 ALGOL 58。在 ALGOL 58 的基础上,1960 年 1 月在巴黎会议上集中了全世界第一流的软件专家讨论确定了程序设计语言 ALGOL 60,发表了“算法语言 ALGOL 60 报告”。1962 年又发表了“算法语言 ALGOL 60 的修改报告”。它是程序设计语言发展史的一个里程碑,程序设计语言由技艺转向科学的重要标志。开拓了程序设计语言的研究领域,又为后来软件自动化的工作以及软件可靠性问题的发展奠定了基础。它的主要特点是:

(1) 局部性 首次引进局部性概念,既扩充了语言的表达能力,又可节省内存空间,提高了程序的紧凑性。

(2) 动态性 语言含有动态成分,从而明显提高了语言的表达能力,不过实现中的开销也明显增加。

(3) 递归性 递归性的引进开拓了软件的研究领域,促进了软件的发展。

(4) 严谨性 它的语法和语义均有严格的描述,特别是,语法的描述采用了特定的形式体系巴科斯形式体系(BNF),结构清晰,理论严谨。

在 ALGOL 基础上发展的 ALGOL 语言家族包括 ALGOL 68(参见 ALGOL 68),Simula 等。经历了 22 年后 ALGOL 60 国际标准于 1984 年公布,编号为 ISO1538:1984,全名为“Programming Language ALGOL 60”。我国 1979 年 12 月发表国家标准(编号为 GB 1500-79),全名为《程序设计语言 ALGOL》,它是我国制订的第一个程序设计语言国家标准。

参考文献

1. 徐家福. ALGOL 漫谈. 南京大学学报(计算机科学专刊二),1979
2. Naur P, et al. Revised Report on the Algorithmic Language ALGOL 60. International Federation for Information Processing, 1962 (郑国梁)

ALGOL 68 yuyan

ALGOL 68 语言 (ALGOL 68 language) 一种

高级程序设计语言,其正式文本“算法语言 ALGOL 68 报告”问世于 1968 年,而最后文本“算法语言 ALGOL 68 修改报告”到 1975 年才发表。两个报告的首席设计人均均为荷兰计算机科学家 A. van Wijngaarden。

ALGOL 68 的语法用一个二级文法表示,称为 W 文法。其中第一级文法描述元语言,第二级文法描述 ALGOL 68 语言本身,但其中含有元语言中的元概念。因此,虽然这两级文法各自都是上下文无关的,它们的组合却可以生成无穷多条语法规则,从而具有很强的上下文有关描述能力。ALGOL 68 的语义用一种部分形式化的英语表示,写成抽象机的形式,属于操作语义。

ALGOL 68 采用正交设计,尽可能使每一种运算能作用于每一种数据类型。它是一种强类型语言。它的类型称为模式。运算过程中允许有限的模式转换,称为强制,所有强制都体现在语法公式中。ALGOL 68 又是一种可扩充语言。通过模式说明、运算说明和(运算)优先说明可以扩充新的模式和新的运算符,但没有数据封装和抽象数据类型。它允许运算符重载,但没有多类型机制。

ALGOL 68 的程序组织采用分程序嵌套结构。除了传统的循环、条件和 goto 语句外,它还引入了平行语句。并发控制采用 PV 操作。输入输出(统称传输)以文件操作为基础,允许用户自定义传输格式。

20 世纪 60 年代中期,国际信息处理联合会(IFIP)为确定 ALGOL 60 的后继公开征求新的语言方案。ALGOL 68 在竞争中获胜,并于 1968 年 12 月在慕尼黑举行的 IFIP2.1 工作小组会议上正式通过公布。但它从一开始就引起异议。著名计算机科学家 E. W. Dijkstra 等人同时公布了他们的“四数报告”,表示坚决反对。反对理由主要是认为该语言文本的晦涩难懂以及该语言无助于提高编程可靠性。后来,ALGOL 68 只在少数国家(主要是西欧)流行,并且未能成为国际标准化组织(ISO)的标准。而当年在与 ALGOL 68 竞争中失败的 SLGOLW 却以 PASCAL 的名字获得了很大的发展。不过 ALGOL 68 含有许多很好的语言设计思想,它们仍是计算机科学的重要成果。

参考文献

1. van Wijngaarden A 等著. 算法语言 ALGOL 68 修改报告. 陆汝钤译. 北京: 科学出版社, 1982
2. 陆汝钤著. ALGOL 68 导引. 电子计算机参考

资料, 1975(69, 70): 1 ~ 178

(陆汝钤)

AO* suanfa

AO* 算法 (AO* algorithm) 一种用于可分解型产生式系统的与或图搜索策略。可分解型产生式系统是产生式系统的一种形式,其综合数据库和结束条件是可分解的。在这类系统中分量数据库在处理过程展开时动态地重新排序,与或图结构则非常适合于描述这种产生式系统。

一个可分解型产生式系统规定了一个隐含的与或图,它由综合数据库标记的一些节点及其后继节点与 k- 连接符组成,当所有连接符为 1- 连接符时,则得到一个普通的图。和与或树不同,在与或图中一些节点既可以相对某一个父节点为与节点,也可以相对另一父节点为或节点,因此在与或图中通常不将节点称为与节点或或节点,而是引入更一般的标记描述与或图。在与或图中,初始数据库对应于初始节点,并通过一个外向连接符与一组后继节点相连,后继节点分别对应于初始数据库的分量(若初始节点可被分解),或应用产生式规则(对应图中一个连接符)后产生的分量数据库。在隐含图中,和满足产生式系统结束条件的数据库相对应的有一组终结节点。通常,与或图搜索的任务可视为寻找从初始节点到终节点的一个解图,即从初始节点开始,正确地选择一个连接符,从该连接符所指的每个后继节点出发,继续选用一个连接符,依此类推,直到由此产生的每个后继节点均为终结节点集里的元素为止。而解图的费用可基于连接符费用来计算,即从任意节点 n 到节点集 N 的一个解图的耗散值记为 $k(n, N)$,并递归定义值 $k(n, N)$ 如下: 若 n 是 N 的一个元素,则 $k(n, N) = 0$; 否则, n 有一个到解图中后继节点集 $\{n_1, \dots, n_i\}$ 的外向连接符,设其费用为 c_n , 则

$$K(n, N) = c_n + k(n_1, N) + \dots + k(n_i, N)$$

AO* 算法是与或图的一种启发式搜索过程,它使用启发函数 $h(n)$ 估价 $h^*(n)$ (n 到终结节点集的最佳解图的费用),并对 h 加以单调限制,使得隐含图中从节点 n 指向其后继节点 n_1, \dots, n_k 的每一连接符均被限制,假设

$$h(n) \leq c + h(n_1) + \dots + h(n_k)$$

其中 c 是连接符的费用。这一限制类似于普通图中对启发函数的单调限制,且扩展一个节点过程就是应用一个后继节点操作符来生成一个节点(通过全

体外向连接符)的全部后继节点。下面我们给出与或图启发式搜索过程 AO^* :

(1) 建立仅含初始节点 s 的搜索图 G , 节点 s 的费用, 若 s 是终节点, 则标记 s 为 SOLUED;

(2) until s 已被标记 SOLUED, do;

(3) begin;

(4) 计算 G 中的一个局部解图 G' : 跟踪 G 中离开 s 的有标记的连接符;

(5) select G' 的任意一个非终叶节点;

(6) 扩展节点 n , 生成它的全部后继节点, 并存于 G 中, 对 G 中未曾出现的每一个后继节点 n_j , 其耗散值 $q(n_j) = h(n_j)$, 若 n_j 是终节点, 则标记为 SOLUED;

(7) 建立仅含 n 的一个单一节点集 S ;

(8) until S 为空, do;

(9) begin;

(10) 从 S 中删除一个节点 m , 使其在 G 中无后裔出现在 S 中;

(11) 修改 m 的费用: 即对从 m 指向节点集 $\{n_{i1}, \dots, n_{ki}\}$ 的每一个连接符计算:

$$q_i(m) = c_i + q(n_{i1}) + \dots + q(n_{ki})$$

令 $q(m)$ 是全体外向连接符 $q_i(m)$ 中的最小值, 并加标记到具有最小值的连接符上。若原标记与新标记不同, 则删除原标记。若该连接符指向的全部后继节点均被标记为 SOLUED, 则标记 m 为 SOLUED;

(12) 若 m 被标记为 SOLUED, 或 m 的修正耗散值不等于原费用, 则将 m 的所有父节点加到 S 中, 使 m 成为它们通过有标记的连接符的后继节点之一;

(13) end;

(14) end。

AO^* 算法包含两个主要的操作: 一个是在第(4)至第(6)步中的自上而下的图生长, 并通过跟踪有标记的连接符寻找最佳解图; 另一个是在第(7)至第(12)步中的自下而上的费用修正、连接符标记和 SOLUED 标记过程。当给初始节点分配一个极高的费用时, AO^* 算法将无法得到该节点的解图; 当令 $h \equiv 0$ 时, 则 AO^* 为一个宽度优先算法; 当从给定节点到一组终结节点存在一个解图时, 且对所有节点有 $h(n) \leq h^*(n)$ (即 h 满足单调限制), 则 AO^* 算法最终总能得到一个最佳解图。

与普通图的算法 A^* 相同, AO^* 可以利用不同的方式修正, 以便得到一些实用于特定情况的算法。目前, 与或图搜索已得到广泛的应用, 如材料切割问

题、动态规划、符号积分及定理证明等领域, 已证明与或图等价于上下文无关文法, 而且当将与或图推广表示非独立的子问题时, 这种广义图与 0 型文法等价。

参考文献

1. Nillson N J 著. 人工智能原理. 石纯一等译. 北京: 科学出版社, 1984

2. Nilsson N J. Problem-Solving Methods in Artificial Intelligence. New York: McGraw-Hill, 1971

3. Martelli A, Montanari U. Optimizing Decision Trees through Heuristically Guided Search. CACM, 1978, 21(12): 1025 ~ 1039 (怀进鹏)

BASIC yuyan

BASIC 语言 (BASIC language) 一种简单易学, 使用方便的交互式语言。BASIC 是 beginner's all-purpose symbolic instruction code (初学者通用符号指令代码) 的缩略语。美国 J. G. Kemeng 和 T. E. Kurtz 于 20 世纪 60 年代初开始研制, 1966 年正式推出。BASIC 最初是为初学计算机的人设计的, 因而简单易学, 小巧灵活, 使用方便, 既可作为批处理语言使用, 又可作为分时语言使用; 既可用解释程序直接解释执行, 也可用编译程序编译成目标代码再执行。BASIC 具有交互会话功能, 在程序执行过程中用户和机器可以相互问答, 并可在程序执行暂停时插入新的语句执行。但 BASIC 不适用于编写较大的程序。

BASIC 程序由若干个相连的语句行组成, 每一语句行的前面可以有一行号 (在较早版本中每一语句行前均需有一行号), 每一语句行包含一个语句 (在有些版本中允许包含多个语句)。各语句行 (按其行号的顺序) 依次执行。但可用控制语句来改变执行流程。语句可分为说明语句, 输入输出语句, 控制语句等几类, 语句语法结构比较简单。数据类型只有简单类型与数组两种。在较早版本中变量名只能是一个字母或一个字母后接另一个字符, 一个程序中最多允许使用 260 个变量名; 数组只能用一個字母表示, 从而最多只能使用 26 个数组。另外, 用户还可以最多定义 26 个自定义函数。

下面是一个求 $N!$ (N 的阶乘) 的程序:

```
PROGRAM FACT
INPUT N
LET F = 1
```

```

FOR I=1 TO N
  LET F=F*I
NEXT I
PRINT N;"的阶乘为";F
END

```

BASIC 问世后,人们对之作多种扩充,在最初的扩充中,吸取了 COBOL 与 ALGOL 等语言的长处。目前的扩充涉及矩阵运算,图形处理,文件处理,字符串处理,以及结构化控制语句等。目前的 BASIC 比最初的基本语言无论在形式上,功能上,还是在规模上都有很大不同。

1978 年美国发布最小 BASIC 国家标准(标准号 ANSI X3.60—78),1984 年该标准上升为国际标准(标准号 ISO 6373—84),1987 年美国发布全 BASIC 国家标准(标准号 ANSI X3.113—87)。我国于 1991 年发布 BASIC 子集国家标准(标准号 GB 12856—91)。

参考文献

1. Kemeng J G, Kurtz T E. BASIC programming. 2nd Ed. New York: John Wiley & sons, 1971
2. 中华人民共和国国家标准,GB 12856—91. 国家标准 BASIC 子集,1991 (徐宝文)

BCY yuyan

BCY 语言 (BCY language) 汉语拼音 bianyi chengxu chushi yuyan(编译程序初始语言)的缩略语。它是一个与算法语言 ALGOL 60 相类似的语言(参见 ALGOL60 语言),于 20 世纪 60 年代初期由中国科学院计算技术研究所的一个小组所设计。

BCY 具有与 ALGOL 60 类似的基本语言成分,但是避免了当时已知存在于 ALGOL 60 中的漏洞。这些与计算工具无关的语言成分有:计算语句、转向语句、空语句、循环语句、条件语句、子程序语句、简变说明、场说明、开关说明和子程序说明等。

BCY 与 ALGOL 60 不同之处有,增加了为描述数字计算机上的计算过程用的其他语言成分,如:输入语句、印刷语句、鼓传送语句、带传送语句、停语句、求和语句、鼓说明、带说明和修改部分等。于是可以描述磁鼓、磁带、输入、输出设备的使用,以及描述在编译前对源程序所作的修改。

BCY 的其他特点包括:使用汉字定界符,如始、终、若、则、否则、转、对于、执行、步长、到、当等共 33 个。BCY 的表达式中还允许使用机器字,并可以对

其中的字段进行运算。

BCY 语言的编译系统首先于 1965 年在中国科学院计算技术研究所的 119 计算机上实现。以后又分别在该所的 109 乙机、109 丙机、015 机,以及电子工业部华北计算技术研究所的 DJS-8 机、华东计算技术研究所的 655 机上实现。(董毓美)

C ++ yuyan

C ++ 语言 (C ++ language) 一种面向对象语言。C ++ 语言最先由 AT&T 公司 Bell 实验室计算机科学研究中心的 B. Stroustrup 在 20 世纪 80 年代初设计并实现,它是以 C 语言为基础的支持数据抽象和面向对象风范的通用程序设计语言,至今仍在进一步演变发展。

C ++ 是对 C 语言的扩充,扩充的绝大部分来自著名语言中的最佳特性:从 SIMULA 67 中吸取了类,从 ALGOL 68 中吸取了算符一名多用、引用和在分程序中任何地方说明变量,综合了 Ada 的类属和 Clu 的模块特点,形成了抽象类,从 Ada、Clu 和 ML 吸取了异常处理,从 BCPL 中吸取了用“//”表示注释。

C ++ 保持了 C 的紧凑、灵活、高效和易移植性强的优点,它对数据抽象的支持主要在于类概念和机制,对面向对象风范的支持主要通过虚拟函数。由于 C ++ 既有数据抽象和面向对象能力,又比其他面向对象语言如 Smalltalk, Eiffel, Commonloop, CLOS 等的运行性能高得多,加上 C 语言的普及,而从 C 至 C ++ 的过渡较为平滑,以及 C ++ 与 C 的兼容程度可使数量巨大的 C 程序能方便地在 C ++ 环境中重用,使得 C ++ 在短短的几年内迅速流行,成为当前面向对象程序设计的主流语言。

C ++ 的标准化工作由美国首先发起。1989 年美国国家标准学会(ANSI)成立了 X3J16C ++ 标准化委员会,1991 年 6 月,以 X3J16 的国际小组为主成立了 C ++ 国际标准化工作组 ISO/IEC JTC1/SC 22/WG 21,两个组织决定联合工作,以一个标准文本同时作为 ANSI 和 ISO 的标准,C ++ 标准化工作划分为核心,扩充,库,环境,与 C 兼容性和国际化等专题小组,预计 1996 年可形成正式标准。

参考文献

1. Stroustrup B. The C ++ Programming Language. (2nd ed). Addison-Wesley, 1991
2. X3J16, SC 22/WG 21: Working Paper for Draft Proposed International Standard for Information Systems — Programming Language C ++ (金益民)

CGI tuxing jiekou biao zhun

CGI 图形接口标准 (Computer Graphics Interface Standard, CGI) ISO/IEC JTC1/SC24 制定的软件与图形设备之间接口的一个国际标准(ISO/IEC 9636),也是软件与图形设备进行对话的一种接口技术。

CGI 描述了图形系统中独立于设备部分和依赖于设备部分之间的接口,其目的是在程序和虚拟设备之间以一种与设备无关的方式提供图形信息的描述和通信。CGI 提供 5 部分功能:①控制、查询及出错处理;②输出图元及其属性;③图段定义及处理;④输入与应答;⑤光栅图像处理。

CGI 的输出图元是组成图片的几何要素。图元都是在虚拟设备坐标空间中定义的。虚拟设备坐标空间是一个二维的笛卡尔坐标系,用户可以使用整数或实数定义一个矩形窗口,并在其中描述所处理的对象。输出时该窗口被映射到物理设备显示面的一个矩形区域中。这样,用户就不必了解设备的准确工作范围和分辨率等参数。另外,CGI 也提供在虚拟设备坐标空间中是否进行裁剪及如何裁剪的控制。

CGI 可以与设备进行对话以了解其功能和参数,使软件能有效使用该设备。CGI 提供了从图形缓冲存储器中输出图形并不断修改图形的手段,也可以强迫设备自动连续修改图形。CGI 还具有软复制设备和硬复制设备的模拟功能。

CGI 把图像定义为矩形的像素阵列,称为**光栅**。CGI 提供光栅的产生、储存、检索和显示的功能,包括光栅控制功能、光栅属性功能、光栅操作功能、光栅询问功能等。

作为与设备无关的一种图形设备的控制标准,CGI 的基本目标是实现高层的图形软件如 GKS 和 PHIGS 等提供有效的支持。当然,CGI 也可以以子程序包的形式直接提供给编程人员使用,以增强应用软件的灵活性和适应性。此外,CGI 也是不同图形设备之间的数据交换标准,图形系统所生成的 CGI 格式的数据流,可以直接驱动能接受 CGI 数据格式的设备,如图形显示器、绘图机等。

参考文献

1. ISO/IEC 9636:1991, Information Technology—Computer Graphics—Interfacing Techniques for Dialogues with Graphical Devices (CGI)—Functional Specification—Part 1 ~ Part 6

2. ISO/IEC 9637:1994, Information Technology—Computer Graphics—Interfacing Techniques for Dialogues with Graphical Devices (CGI)—Data Stream Binding—Part 1 ~ Part 2 (张福炎)

CIP-L yuyan

CIP-L 语言 (CIP-L language) 一种支持转换式程序设计的广谱语言。CIP-L 可描述从软件功能规约、设计规约、作用式程序到过程式或面向机器的高效程序等不同的抽象级;提供了从高抽象级向低抽象级转换的语言结构和语义规则。它是 70 年代中期由德国慕尼黑技术大学 F. L. Bauer 主持开发的“直觉引导的计算机辅助程序设计”项目的主要部分,用以支持转换式程序设计方法(参见**转换方法**),保证程序转换的正确性。

作为广谱语言,CIP-L 用代数类型描述功能规约,用计算结构刻画代数类型的模型,用模式描述算法和控制结构。代数类型也就是**抽象数据类型**,其公理用一阶逻辑公式表示。CIP-L 要求,代数类型中的任一对象均可经有限步基本运算生成。这种生成结构可用 CIP-L 的计算结构刻画。模式是一个算法架构,用以描述算法策略和控制策略,将一个模式中的代数类型名赋予具体的语义解释后即可得到具体的算法。模式语言又分成核心层和多个扩展层。核心层定义了表达式子语言,可按逻辑式或函数式风格书写程序,采用模型论方法定义其语义。扩展层包含了说明性子语言(参见**说明性语言**),命令式子语言(参见**命令式语言**),并行子语言和控制子语言,它们的语义刻画采用转换语义方法,即每一语法结构的语义均可由转换规则在有限步内归结到核心层语义。

CIP-L 针对转换式程序设计的需要而设计。语言覆盖了从功能规约到过程实现各个层次,功能很强,但同时造成结构繁杂,不易掌握。

参考文献

Bauer F. L. et al. The Munich Project CIP: The Wide Spectrum Language CIP-L. Berlin:Spring-Verlag, 1985 (伊波)

COBOL yuyan

COBOL 语言 (COBOL language) 一种用于事务处理的通用程序设计语言。COBOL 是 common business oriented language 的缩略语。

1959年5月由美国政府部门、用户、制造商和其他部门参加的会议上决定成立 CODASYL, 并提出了用于事务数据处理的语言需求。要求面向用户, 面向问题, 与机器无关, 要求用简单英语或类英语表示, 并尽可能避免符号化。

1959年9月由 CODASYL 的一个委员会提出了 COBOL 初稿, 其最终报告被采纳并于 1960 年 4 月公布, 称为 COBOL 60。

COBOL 语言是最早出现的程序设计语言之一。最早应用于事务数据处理, 并使用文件、记录、组项、初等项的数据描述方法。早期使用十分广泛。标准化活动开展最早且至今仍有活力。通过不断吸收新概念, 采用兼容式的发展和“过时”元素的处理方法来不断更新语言版本, 以适应事务数据处理领域的各种新需求。

COBOL 语言将其核心和功能模块均分成若干级, 故可以灵活地构成功能上具有积木式的但又是标准的适用于不同层次的语言实现系统, 从而适应不同的需求。

COBOL 程序由 4 个部组成。标识部用于刻画程序的标识性特征; 环境部用于刻画程序和计算机环境有关的成分; 数据部用于刻画数据(包括文件、记录、组项和初等项等)的定义的构成以及相关特征和属性; 过程部用于刻画处理加工流程, 采用节、段、语句的结构层次。整个程序具有类英语的描述特点, 具有较好的易读性。

目前的 COBOL 国际标准版本是 ISO COBOL 1989—1985。其第一版为推荐版 ISO COBOL R-1989—1972, 它由 1 个核心模块和 7 个功能处理模块(表处理、顺序存取、随机存取、分类、报表编制、程序分段和库)组成。其第二版为标准版 ISO COBOL 1989—1978, 它由 1 个核心模块和 11 个功能处理模块(表处理、顺序输入输出、相对输入输出、索引输入输出、分类合并、报表编制、程序分段、库、排错、程序间通信和通信)组成。其第三版即为 ISO COBOL 1989—1985, 它由 7 个必需模块(核心、顺序输入输出、相对输入输出、索引输入输出、程序间通信、分类合并、源正文管理)和 4 个任选模块(报表编制、通信、排错、程序分段)组成。

我国的 COBOL 国家标准采用相应的国际标准。

参考文献

1. 中华人民共和国国家标准 程序设计语言 COBOL (GB 4092—83) 北京: 中国标准出版社, 1985

2. ISO Standard 1989—1985 (American National Standard COBOL X3.23-1985) (钱树人)

C yuyan

C 语言 (C language) 一种使用颇为广泛的程序设计语言。它由 AT&T 公司 Bell 实验室的 D. Ritchie 于 1972 年至 1973 年间在类似于 BCPL (由英国剑桥大学的 M. Richards 于 1969 年设计并实现的系统程序设计语言) 的 B 上设计而成, 故命名为 C。首先在 DEC PDP-11 机上实现, 著名的 Unix 操作系统就是用 C 书写的。

C 语言在很多方面继承和发扬了 20 世纪 60 年代出现的许多高级程序设计语言的成功经验和特色, 它使用自由书写格式, 具有丰富的数据类型, 多种存储类别, 一定程度的模块化结构, 采用结构化的控制, 函数参数传值, 并支持分别编译等。主要特点是语言与运行支撑环境分离, 语言规模小, 相对简单, 表示方法简洁, 高度灵活, 程序运行效率高, 可移植性好; 有不少操作直接对应于实际机器所执行的动作, 在许多场合可代替汇编语言; 大量使用指针, 对运算时数据类型的一致性限制较少。

尽管最初是作为一种系统程序设计的工具语言设计的, 但 C 已成功用于各个应用领域, 是当前使用最广泛的一种通用程序设计语言。

1983 年美国国家标准学会 (ANSI) 的 X3J11 委员会开始进行 C 的标准化工作, 国际标准化工作始于 1985 年 (ISO/IEC JTC 1/SC 22 WG 14), 1990 年公布的国际标准 ISO/IEC 9899 以美国国家标准 ANSI C 为基础, 是第一个支持多八位字符集的程序设计语言国际标准。我国国家标准等同采用了 ISO/IEC 9899。为使 C 语言能更好地支持对世界各国语言文字的处理, 适应新的编码字符集国际标准 ISO 10647, 以及进一步发展 C 语言, ISO/IEC JTC 1/SC 22 WG 14 还在继续工作, 预计 1995 年可正式公布 C 国际标准的补篇。

参考文献

1. Kernighan B W, Ritchie D M. The C Programming Language (2nd ed.). Englewood Cliffs: Prentice-Hall, 1988

2. ISO/IEC 9899: 1990 (E) International Standard for Information Systems — Programming Language C. ISO/IEC Copyright Office, 1990 SC 22/WG 14/N288; ISO/IEC 9899: 1990/Amendment 1: 1994 (E)

(金益民)

DOS caozuo xitong

DOS 操作系统 (DOS operating system)

用于微型计算机的一种磁盘操作系统。

DOS 是美国 Microsoft 软件公司与 IBM (国际商业机器公司) 开发的, 广泛运行于 IBM PC 及其兼容机上的磁盘操作系统, 全名叫 MS-DOS。

20 世纪 80 年代初, IBM 公司开发 IBM PC, 当其涉足微型计算机市场时, 曾多方考察选择配合该机的操作系统, 1980 年 11 月, IBM 和 Microsoft 正式签约, 日后的 IBM PC 均使用 DOS 为标准的操作系统。由于 IBM PC 大获成功, Microsoft 也跟着得到了飞速发展, MS-DOS 从此成为个人计算机操作系统的代名词, 发展成个人计算机的标准平台。

在 IBM PC 机上所配的操作系统称 PC-DOS 或 IBM-DOS, 是由 IBM 向 Microsoft 买下 MS-DOS 的版权, 另外作了修改和扩充而产生的。

MS-DOS 最早的版本是 1981 年 8 月发表的 1.0 版, 至 1993 年 6 月推出了 6.0 版。MS-DOS 是一个单用户微型计算机操作系统, 4.0 版开始具有多任务处理能力。MS-DOS 的主要功能有: 命令处理、文件管理和设备管理。命令处理对用户输入的键盘命令进行解释和处理; 文件管理负责建立、删除和读写各类文件; 设备管理完成各种外围设备, 如键盘、显示器、打印机、磁盘和异步通信设备的输入输出操作。此外, MS-DOS 还具有系统管理和内存管理等功能。

MS-DOS 采用分层模块结构, 按照功能划分, 它由组成层次的 4 个模块组成, 其结构如图 1 所示。

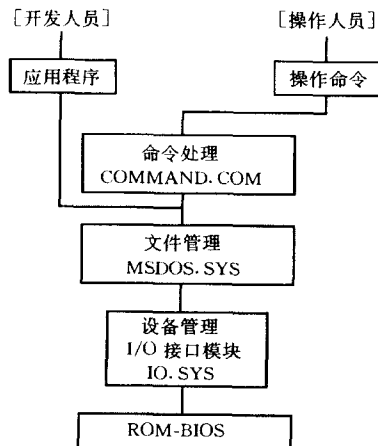


图1 MS-DOS分层模块结构框图

(1) 基本输入输出系统 ROM-BIOS 存放在只读存储器中, 它提供对 PC 机的 I/O 设备, 如显示器、磁盘驱动器等最基本的 I/O 操作服务。ROM-BIOS 位于 DOS 的最底层, 直接和硬件设备交互。它自身又由加电自测程序、I/O 支撑程序等组成。

(2) 输入输出接口模块 IO.SYS 是 MS-DOS.SYS 和 ROM-BIOS 的接口模块, 它与 ROM-BIOS 共同处理 I/O 操作, 统称为设备管理。其主要任务是: 测定系统状态并进行系统初始化, 管理和驱动各种外围设备, 使磁盘系统复位, 为引入内存的 MS-DOS.SYS 重定位。

(3) 文件管理模块 MS-DOS.SYS 是 MS-DOS 操作系统的核心部分, 它提供系统与应用程序间的接口, 其主要任务是: 文件管理, 存储器管理和提供例行程序服务。

(4) 命令处理模块 COMMAND.COM 位于 MS-DOS 的最上层, 直接和操作人员打交道, 接收从键盘来的输入命令, 并确定如何处理这些命令。

MS-DOS 中所有的信息都以文件形式存储在磁盘上, 每个文件都有惟一的名称, 以供识别。文件的名称由两部分组成: 文件名和扩展名。前者由 1 至 8 个字符组成; 后者是以圆点开始的, 可取 1 至 3 个字符。文件名和扩展名可用的字符有: \$, #, &, @, !, %, (, \), |, \, ^, ~ 等。文件的扩展名用于对文件进行分类, 补充说明文件的特性。MS-DOS 中, 有一批约定使用的扩展名, 例如: .COM (命令文件), .EXE (执行文件), .BAT (批文件), .SYS (系统文件), .BAK (后备文件), .LIB (库文件) 等等。

文件目录是用来实现“按名存取”的主要手段, 每个文件都有一个文件目录项, 而文件目录项的集合就构成了文件目录。MS-DOS 的文件目录项由 32 个字节组成, 如图 2 所示, 其中包含了该文件的主要属性。

起始字节	字节个数	说 明
0	8	文件名
8	3	扩展名
11	1	文件可访问性
12	10	保留专用
22	2	建立或修改时间
24	2	建立或修改日期
26	2	起始簇号
28	4	文件字节数

图2 MS-DOS的文件目录项

其中, 文件可访问性区分是: 读写文件、只读文件、隐

含文件、系统文件或子目录信息等。起始簇号指明该文件的信息在磁盘空间中的第一个信息块的位置,1个盘簇等于2个至64个扇区不等,每个扇区存放512个字节。建立文件时,MS-DOS自动在磁盘目录区建立这个文件的文件目录项。采用树型目录结构组织系统中所有文件的文件目录项,树的结点分3种:根结点、枝结点和叶结点。根结点表示根目录,枝结点表示子目录,叶结点表示文件目录项,亦即代表了这个文件。树中每个结点有一个名字以供访问,根目录是惟一的,其名字为反斜杠“\”;子目录和文件的名字可由用户或系统给定,只要符合文件或子目录的命名规则即可。每个子目录均对应一个子目录项,其内容和文件目录项相似,包含了子目录的若干重要属性。

每个磁盘(软磁盘或硬磁盘)只有一个根目录,在磁盘初始化时由系统自动建立。根目录用来存放根目录下的所有文件和子目录的目录项,不同类型磁盘的根目录区大小不同,可从4个至64个扇区不等,因而,根目录中可容纳的目录项数就不同。由于根目录中可存放文件目录项,也可存放子目录项;子目录中可存放文件目录项,也可存放子目录项,而子目录中包含的目录项信息都存放在用户空间,这样,就形成了一个树型目录结构,只要有足够的存储空间,文件和子目录的个数就不受限制。

对于树型结构的文件目录,为了查找一个文件或子目录,必须给出这个文件或子目录的路径。路径是从指定目录或当前目录开始,到达指定文件或子目录所经过的路线。路径可以从根目录开始,也可以从当前目录开始,并由经过的子目录名组成,子目录名间用“\”分隔,如果路径末尾跟有文件名,也用“\”将文件名和它前面的目录名分开,这样形成了一条路径。

MS-DOS按用户指出的路径来查找文件或子目录。以“\”开头的路径叫绝对路径,DOS就从根目录开始查找,否则就表示相对路径,DOS就从当前目录开始查找。

MS-DOS的文件引用名可以看作是文件名的扩展,它是在文件名的前边加上盘符和路径,能进一步指明文件或子目录的具体位置,例如,A:\XSDOS\WPS.EXE,B:\FoxBASE\Fox.EXE,...\LCH.COM,C:\ABC\BIN等都是文件引用名的例子。

MS-DOS的文件名字中可以使用通配符。通配符“?”代表文件名或扩展名中该符号所在位置的任一可用字符。通配符“*”代表文件名或扩展名

中该符号所在位置的任一字符串。由于使用通配符的文件名字能代表一类文件,就可一次复制、删除或操纵一批文件,减少击键次数,提高操作效率,方便用户使用。

MS-DOS文件的存取是由文件目录和文件分配表配合完成的。使用格式化命令格式化磁盘时,磁盘空间划分成5个部分:引导记录,FAT1,FAT2,根目录区和用户数据区。其中,文件分配表FAT是文件管理的重要数据结构,用以记录磁盘文件空间的使用情况。FAT2是FAT1的复制品,以防FAT1被破坏而设置。FAT的大小可变,例如,3.5英寸1.44MB盘,两个FAT每个占9个扇区;对大容量硬磁盘来说,每个FAT占256个扇区。FAT的每个表目对应一个盘簇,当磁盘容量超过10MB时,表目长16位。每个表目的状态指示它所对应的盘簇为:未分配、已分配、坏盘簇。FAT记录了磁盘空间的使用情况及每个文件的数据占用磁盘空间的位置,由于每个表目对应一个簇,由簇号经过简单换算就可得到扇区号。每个文件占用的FAT表目采用连接方式链起来。因此,从该文件的文件目录项的起始簇号开始就可以最终实现文件信息的读写。

键盘命令是操作员从键盘上输入的,要求MS-DOS完成一定功能的会话命令。MS-DOS的命令分成两类:

(1) 内部命令 直接嵌入MS-DOS的系统驻留区中,执行时不占用内存用户区,一经接收,就能立即执行。

(2) 外部命令 存放在磁盘上的各种命令文件,每当执行时,从磁盘上调到内存用户区,MS-DOS把具有扩展名为.COM,.EXE和.BAT的文件都视作外部命令。

MS-DOS命令可分成以下几类:基本DOS命令、文件操作命令、目录管理命令、批处理命令、高级DOS命令等。

汉字磁盘操作系统(CC-DOS)是为了使原来只具有西文处理能力的PC机DOS系统同时又具有处理中文信息的能力,在1983年,由我国电子工业部第六研究所,在MS-DOS的基础上,专为PC机开发成功的汉字操作系统。CC-DOS的开发和使用,为在我国更广泛地普及和使用微型计算机打下了坚实基础,具有很重要意义。

MS-DOS和CC-DOS的根本区别在于是否支持汉字字符处理,CC-DOS在MS-DOS的基础上工作,除保留原来MS-DOS的几乎全部功能以外,又

增加了:汉字输入、汉字存储和处理、汉字输出功能。

汉字输入是汉字处理技术中最关键又最困难的部分,CC-DOS 提供了面向用户的多种输入方法。面向一般用户的简单输入法有:拼音码、首尾码、五笔字型输入等;面向专业操作员的快速输入法有:国标码、区位码、电报明码等。

汉字字符在计算机中以变形国标码来存储,需用双字节,为了与西文 ASCII 码相区别,其中每个字节的最高位均为 1。CC-DOS 把汉字当作西文字符一样进行处理,汉字可使用到文件的各级,汉字可作为文件名和命令名。各种程序设计语言和应用程序中,汉字可作为变量名、字符串并和西文字符混杂处理。

汉字输出采用字形码,CC-DOS 包含有一个汉字字库 CCLIB,存放不压缩字形码,16×16 点阵汉字字形码使用 32 个字节保存一个汉字的字形,含有 GB2312—80 规定的 6763 个汉字和 682 个图形符号。16×16 点阵字形主要供屏幕显示,每个汉字的高度和宽度比 ASCII 字符大一倍。为了改善字形,又提供 24×24 点阵或 48×48 点阵字形码,由于点字密,字形更为美观,但点阵占用存储空间多。

除了汉字库 CCLIB 外,CC-DOS 还有两个核心文件,以 CC-DOS 4.0 为例,分别称 FILE1.EXE 和 CCCC.EXE。FILE1.EXE 的功能是做好装入汉字库 CCLIB 的准备工作,检查汉字库的完好性,为汉字库申请内存区域,初始处理及进行模式转换。CCCC.EXE 的功能是将汉字库装入申请好的内存区域,完成把 ROM-BIOS 改造成 CC-BIOS,使其能支持各类 I/O 设备的汉字输入输出,再配上原有的 MS-DOS 操作系统,就可以使用键盘、显示器进行汉字的输入输出了。

在 CC-DOS 控制下工作时,有三种操作方式:纯西文方式、中西文方式和纯中文方式。可以通过键盘控制键控制操作方式的切换。

MS-DOS 在 20 世纪 90 年代还在继续发展,1991 年推出 DOS 5.0 版,1993 年 6 月推出 DOS 6.0 版。DOS 5.0 增加了改进的图形用户界面 DOS SHELL;扩大了内存管理能力,可利用 640 kB 以上高地址区存放部分系统驻留程序;提供多任务处理能力,使 PC 能运行前后台任务;直接支持高达 2 GB 的硬磁盘分区;CPU 扩充了保护模式,通过扩充和扩展内存驱动程序,使 PC 286 可访问 16 MB 内存,PC 386 以上可访问 4 GB 内存;增加了 DOSKEY,SETVER,HELP 等 10 多条新命令;支持 3.5 英寸 2.88 MB 软磁盘。DOS 6.0 具有 DOUBLESPACE 压

缩磁盘功能;配置 CD-ROM 驱动程序;帮助设施改为全屏幕交互工作方式;具有 CONFIG.SYS 多重配置功能;进一步改进和增强内存管理功能;增加了消除磁盘碎片及删除整个子目录命令;增加 INTERLNK,使两台微型计算机能直接相连并共享磁盘和打印机;MSAV 工具用来检测和消除病毒;MSD 工具能报告机器硬件及内存工作状态;POWER 工具用来降低程序和机器处于等待态时的功耗。

参考文献

1. Microsoft MS-DOS Operating Version 4.0, Microsoft Corporation, Document Number 410630001-400-R10-1088, 1988

2. Norton P. Inside The PC. Brady, A Division of prentice Hall Computer Publishing Inc., 1993

(费翔林)

Eiffel yuyan

Eiffel 语言 (Eiffel language) 一种面向对象程序设计语言。它由美国交互软件公司 B. Meyer 等人于 1985 年设计,1986 年成为软件产品。

基本要素 包括对象、类、继承和实体。

对象 类的实例,在系统运行期间占有一定存储空间。对象的属性域可以是简单的(如数值、字符等),也可以是对其他对象的引用。对象在系统运行期间动态创建与消亡。

类 程序的惟一构造单位,既是模块,也是类型。Eiffel 程序是类的结构化集合,无主程序概念。Eiffel 语言中严格区分了静态的类和动态的对象:类是一组相似对象的抽象正文描述,任何对象都是某个类的实例。类的特征分为属性和程式:属性表示对象状态,程式是作用于对象属性上的操作。特征可以说明为延迟的,其具体实现由后继类给出。含有延迟特征的类称为延迟类。延迟特征虽无实现部分,但其功能可用语言中提供的断言机制进行描述,延迟类的抽象语义性质可由类不变式表述。Eiffel 还提供了类属机制,允许类有表示类型的一个或多个类属参数。

继承 类间的基本关系。Eiffel 语言支持多继承,引入了特征重定义及重命名等设施。

实体 类正文中的标识符,表示在运行时刻的可能引用。实体是比变量更一般的概念,包括程式的局部变量、程式参数、类的属性和用预定义 RESULT 表达的函数结果标识符等。

基本成分 包括类型、表达式、语句。

类型 Eiffel 是强类型语言,允许完全的静态类型检查。每个实体必须有类型说明,并可进行初始化。简单的预定义类型包括 BOOLEAN(布尔型)、CHARACTER(字符型)、INTEGER(整型)和 REAL(实型)。类可用作类型,它是用户自定义类型的设施。为保证同类型的变量在其后继类中保持一致,Eiffel 语言引入了 LIKE(依存类型),即若说明 $x \text{ LIKE } y$,则 x 的类型始终保持与 y 的类型相同。

表达式 包括常量、实体、函数调用、CURRENT、带运算符的表达式等。函数调用形式为 $x.f(p_1, p_2, \dots, p_n)$,它表示调用与实体 x 相关的对象中的函数 f , $p_i (i=1, \dots, n)$ 为函数参数。保留字 CURRENT 本身为表达式,它表示类的当前实例。

语句 包括过程调用、赋值、条件、循环、检查、排错等。过程调用与函数调用形式相同。

实现环境 目前 Eiffel 可在 UNIX 版本下运行,并正被移植到其他环境。Eiffel 环境提供了两个编译命令:EC 和 ES,分别用于编译单个类和整个系统。编译程序以 C 作为中间代码,提供了丰富的编译开关,用以控制系统的运行模式。它还提供虚拟存储机制,支持自动存储管理。Eiffel 的基本类库提供了大量描述常用数据类型的类,如数组、表、堆栈和树等。此外,它还提供了程序调试、文档编制和图形设计等工具。

目前,已有 Eiffel 语言的非盈利国际协会(NICE),负责 Eiffel 语言的标准化和改进;以及 Eiffel 用户小组,负责 Eiffel 的推广应用。

参考文献

1. 徐家福等. 对象式程序设计语言. 南京: 南京大学出版社, 1992
2. Meyer B. Eiffel; The Language. Prentice Hall,

1992

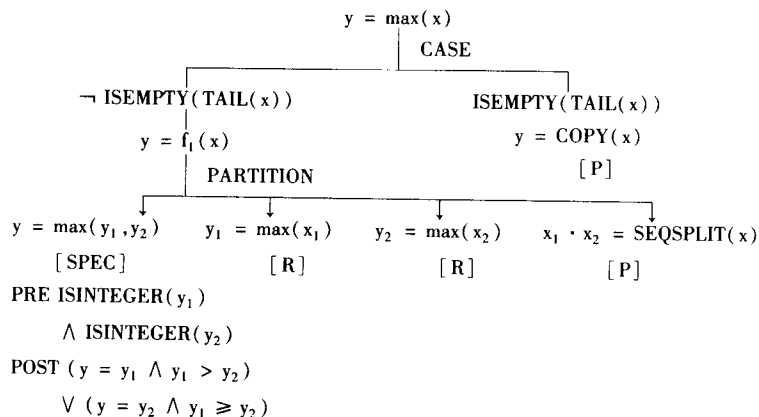
(张家重 王志坚)

FGSPEC yuyan

FGSPEC 语言 (FGSPEC language) 一种基于功能分解的形式化功能规约语言。FGSPEC 用函数作为软件的数学模型,从给定的函数功能规约开始,按某一控制结构将它分解成若干子函数功能规约,直到子函数可用抽象数据类型上的运算实现为止。这样,软件功能规约、设计规约及其间的转换统一在树形结构上。

FGSPEC 是 functional graphical specification 的缩略语。它是 20 世纪 80 年代后期由南京大学计算机软件研究所在徐家福教授主持下开发的。主要目标是研究从功能规约到设计规约的自动转换。它已于 1988 年在 SUN Sparc 工作站上实现。

FGSPEC 用前后断言方法描述功能规约,基本控制结构有 PARTITION 和 CASE。前者将一个功能分解成若干个相关的子功能,其间的相关性由相关变量确定;CASE 则提供条件选择机制,为保证最终程序的确定性,各分支上的条件表达式必须互斥,且所有条件之并是永真式。FGSPEC 提供了抽象数据类型定义设施,用以刻画分解树叶结点上的功能操作。在定义抽象数据类型的公理方面,FGSPEC 允许在等式右方使用 EXIST, ALL, FIND 和 FIND ALL 四个功能描述成分,以增强描述能力。为了直观,它的规约采用树形图表示。下面是用 FGSPEC 写的求整数序列最大元的规约。其中,叶结点下()中的 P 表示语言直接支持的原始结构, R 是递归结构, SPEC 是用前后断言定义的子功能规约。该规约实际上刻画了求整数序列最大元的一个算法。



参考文献

徐家福等. 软件自动化. 北京: 清华大学出版社, 1994 (伊波)

FORTRAN yuyan

FORTRAN 语言 (FORTRAN language)

一种面向过程的程序设计语言。FORTRAN 是 formula translation(公式翻译)的缩略语。

FORTRAN 语言是在 20 世纪 50 年代中期由美国 IBM 公司的 J. Backus 领导的小组为 IBM704 计算机设计的。第一个 FORTRAN 语言标准称为 FORTRAN 66, 在 70 年代修订为 FORTRAN 77, 分全集和子集。1991 年国际标准化组织又批准新的 FORTRAN 标准, 称为 FORTRAN 90(规定 F 后面的 6 个字母用小写)。它是国际上第一个支持多字节字符集的标准, 该标准采纳了我国 FORTRAN 工作组关于 CHARACTER(KIND = , ...) 的建议。

FORTRAN 语言包括常数、变量、数组、算术表达式、逻辑表达式等, 语句分成赋值语句、输入输出语句和格式语句、控制语句、说明语句以及子程序等几类, 详细内容请参见语言标准文本和有关系统的语言基准手册。

FORTRAN 语言主要用于数值计算, 由于 IBM 公司在 FORTRAN 语言诞生不久, 就为计算机配置了 FORTRAN 编译程序, 别的厂商也如此, 所以使 FORTRAN 很快普及, 又由于 FORTRAN 语言标准化工作更促进它的推广应用。FORTRAN 语言的特点是接近数学公式, 简单易用, 功能逐步扩大, 如允许复型与双精度浮点运算, 子程序定义机制, 输入输出的格式说明, 允许布尔表达式, 函数和子例程名可以作为参数传递。FORTRAN 77 扩充了字符处理功能, 使之能应用于非数值运算领域, 还增加了块 IF 语句, ELSE 语句和 END IF 语句等, 使写出的程序趋于结构化, 易读性强。

Fortran 90 又对 FORTRAN 77 作了许多扩充和改进。

- (1) 数组运算机制;
- (2) 改善了数值计算;
- (3) 数据类型参数化, 允许使用多种字符类型, 满足各国字符处理的需要;
- (4) 从 6 种内部数据类型中派生出用户定义的数据类型;
- (5) 模块化数据与过程定义机制, 提供了一种数据与过程包装的强有力的而又安全的形式;

(6) 指针机制, 允许创建和操作动态数据结构;

(7) 增加自由形式的源程序形式;

(8) 提供了过程的递归调用机制;

(9) 提供了附加的控制结构, 如 **do...end do**, **do while** <Condition> 等。

由此可以看出, Fortran90 已经是具有强大数值计算能力的现代高级语言, 程序的书写更趋结构化, 模块化。

现在, 国际 FORTRAN 工作组又提出为适应 ISO/IEC 10646 的颁布, 要制定更新的 FORTRAN 国际标准。另外, 随着计算机科学技术的飞速发展, 超级计算机已经进入了向量处理和并行处理时期。作为科学计算的主流程序设计语言, 扩充 FORTRAN 90 使之提供向量和并行处理功能, 已是其发展的主要趋势, 高性能 FORTRAN (HPF) 正在设计中。HPF 的目标是支持并行程序设计; 能在 SIMD 或 MIMD 机上获得较高性能; 便于在不同体系结构的计算机间移植其目标代码, 主要扩充数据分布特性和并行语句。

参考文献

1. 全国信息技术标准化技术委员会程序设计语言分技术委员会 FORTRAN 工作组. 标准 FORTRAN 90 语言程序设计. 北京: 学苑出版社, 1994

2. Adams J C, Brainerd W S, Martin J C, et al. FORTRAN 90 Handbook. New York: McGraw Hill, 1992

3. Brainerd W S. The Programming Language Standards Scene, Ten Years on FORTRAN. Computer Standards and Interfaces, 1994, 16 (5 and 6): 459, 464 (程虎)

FP yuyan

FP 语言 (FP language) 一种无副作用的具有组合子风格的函数式程序设计语言。它是由 John Backus 在 1977 年接受 ACM Turing 奖的讲演中提出的。FP 程序是没有变量的函数。FP 表达式的计值是在函数一级的运算, 它将函数型 (又称组合型) 施于函数以产生新的函数, FP 的算子具有较强的代数性质, 因此可把程序作为代数项处理。下面的 FP 程序计算二向量的内积:

Def IP = (/ +) ◦ (α ×) ◦ Trans

其中 /, ◦ 和 α 分别表示组合型“插入”, “复合”和“作用于所有”, 它们将已知函数组合成新函数。FP 的特点是:

- (1) 引用透明,程序紧凑;
- (2) 便于表示递归函数,因而,具有较好的表达可计算函数的能力;
- (3) 具有潜在的并行性,便于表示并行算法;
- (4) 具有良好的代数性质,因而,便于使用机械方法来理解和证明程序。FP有多种实现和扩充,最典型的是J. Backus后来提出的FL语言,它是强类型语言并允许高阶函数和用户自定义类型。

参考文献

Backus J. Can Programming be Liberated from The Von Neumann Style? A Functional Style and its Algebra of Program. CACM, 1978, 21(8): 613 ~ 641

(黄林鹏 孙永强)

GKS tuxing biao zhun

GKS 图形标准 (Graphical Kernel System Graphics Standard) ISO/IEC JTC1/SC24 制定的图形软件国际标准。又称图形核心系统,简称GKS。GKS是一个为应用软件服务的基本图形系统,它提供了在应用软件和图形输入输出设备之间的功能性接口,该功能性接口包括在各式各样的图形设备上为交互的或非交互的二维作图所需的全部基本功能:输出功能、输入功能、控制功能、变换功能、图段功能、元文件功能、询问功能和出错处理功能。它由两部分组成,一部分是它的功能描述(ISO/IEC 7942);另一部分是它与FORTRAN, PASCAL, Ada等程序设计语言的联编規約(ISO/IEC 8651)。图1是GKS与应用软件、图形设备之间的关系。

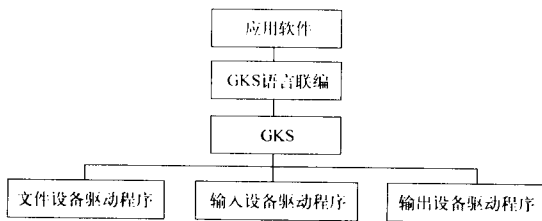


图1 GKS与应用软件、图形设备的关系

GKS是一个通用的综合性的图形标准,它具有二维图形应用程序所需要的大部分功能。然而,对于某些比较简单的应用程序来说,GKS显得过于复杂。此时只要提供GKS的一个子集即可满足要求。为此,GKS在实现时采用分级结构:输入功能分成a,b,c三级,输出功能分成0,1,2三级,相互组合一

共有9级,它们是10a, 10b, 10c, 11a, 11b, 11c, 12a, 12b, 12c,不同级别的GKS可以满足不同应用的需要。

为了满足三维图形应用的需要,ISO还制定了GKS-3D图形标准。GKS-3D是GKS图形标准的三维扩充,它与GKS完全兼容,即所有使用GKS开发的应用软件不加修改就能在GKS-3D上运行。为此,原有GKS的二维功能均未作修改,所有新增加的功能都用于三维处理,且一般都是原来二维功能的扩充和引伸。例如,GKS-3D比GKS增加了6种输出图元,即三维折线、三维标记、三维文字、三维填充区域集、三维彩元阵列和三维通用图元。又如,GKS-3D为三维坐标的输入提供了三维定位和三维笔画两种逻辑输入设备,它们分别是GKS中二维定位设备和笔画设备的引伸。

GKS-3D只是GKS的一种最简练、最必要的扩充,它仅仅包含了三维图形显示技术中最常用的一些基本功能。处理更复杂的图形数据结构和提供更多样三维图元(如定义自由曲面和实体等)的高性能的三维图形软件标准可参见PHIGS图形标准和OpenGL图形标准。

参考文献

1. ISO/IEC 7942: 1994, Information Technology—Computer Graphics and Image Processing—Graphical Kernel System (GKS) Part 1 ~ Part 4
2. ISO 8651: 1988, Information processing systems—Computer graphics—Graphical Kernel System (GKS) Language Bindings Part 1 ~ Part 4
3. ISO 8805: 1988, Information Processing Systems—Computer Graphics—Graphic Kernel System for Three Dimensions (GKS-3D) Functional Description

(张福炎)

Gopher fuwu

Gopher 服务 (Gopher Services) 因特网上提供的一种基于选单驱动的文本信息检索服务。它采用客户-服务器模式,允许用户访问存放在网上的多个Gopher服务器中的信息。

Gopher服务器的内容按目录方式组织,在一个层次结构的选单系统上显示出来。用户只要选中一个主题,就能在一级级选单的引导下,逐步深入。如果发现有价值的信息,可将它取到本地磁盘中,以文件方式保存。

Gopher协议运行在TCP/IP协议之上,其协

议描述在 RFC 1436 文档中定义。Gopher 客户打开到 Gopher 服务器的 TCP 连接,并对其发送请求信号,服务器根据请求信号回送一批数据,并关闭连接。

Gopher 已在因特网上广泛使用,已设置的 Gopher 服务器超过 5 000 个。美国明尼苏达大学是 Gopher 的发源地。
(胡道元)

Gouraud ming'an chuli

Gouraud 明暗处理 (Gouraud shading) 对曲面物体或多边形网格的光强采用插值方法进行明暗处理的一种过程和方法。在应用问题中,曲面物体常常近似地表示为一组多边形(四边形)小面片,这样每个面片的法向量便是常量。在使用扫描线方法对这些面片进行绘制时,Gouraud 明暗处理方法是沿扫描线对光强作线性插值,而不是如 Phong 明暗处理方法那样对法向量进行线性插值。该算法的具体处理过程类似于 Phong 明暗处理方法:首先在多边形顶点处求法向量(其法向量通常用围绕顶点周围的多边形法向量的平均值来表示),然后使用局部光照计算模型(参见**光亮度计算**)计算顶点处光强。多边形内扫描线上各像素点处的光强直接由多边形顶点处的光强经双线性插值得到。

Gouraud 明暗处理方法直接对光强值而不是对法向量进行插值,因而它比 Phong 方法计算量小,简单实用。为此,该方法得到更广泛的应用。在许多高性能的图形工作站中,该方法作为图形加速器或基本图形软件业已实现的功能,可提供给用户直接使用。

参考文献

1. 唐荣锡,汪嘉业,彭群生,汪国昭等. 计算机图形学教程(修订版). 北京:科学出版社,2001
2. 唐泽圣,周嘉玉,李新友. 计算机图形学基础. 北京:清华大学出版社,1995 (吴恩华)

GSPEC yuyan

GSPEC 语言 (GSPEC language) 图形化的设计规约语言。将软件功能树形分解和抽象数据类型有机结合起来,用于描述形式软件设计规约。GSPEC 是 graphical specification 的缩略语,由南京大学计算机软件研究所徐家福教授等人于 1987 年提出。

GSPEC 由**抽象数据类型**机制和软件功能分解

机制两部分组成。抽象数据类型机制包括抽象数据类型定义机制 (DTYDE) 和数据类型实例化机制 (INSTANTIATION)。数据类型实例化可用于有参抽象数据类型定义和语言中定义的复合类型的例化。抽象数据类型定义机制由基于子句 (BASED ON)、引进子句 (INTRODUCE)、数据说明子句 (DECLARE)、公理子句 (AXIOM) 组成。基于子句用于刻画多个数据类型之间的层次式依赖关系,引进子句刻画抽象数据类型中所定义操作的语法形式,即型构,数据说明子句对公理子句中出现的量作类型说明。公理子句用代数和一阶谓词相结合的方法,通过操作的性质和各操作间的关系来刻画抽象数据类型中所定义的操作的含义。每条公理的一般形式为右部可以带有条件的等式,并且等式右部还可以出现 EXIST (存在量词)、ALL (全称量词)、FIND (找出满足后面公式的任一值),和 FIND ALL (找出满足后面公式的所有值的集合) 这四种功能性描述成分。

GSPEC 语言的软件功能分解机制提供了任意有限多叉的树形分解模式,形象直观,层次分明。该语言成分有三种:基本控制结构,控制结构定义和函数定义。基本控制结构有 PARTITION 和 CASE。控制结构定义由控制结构分解树 (STRUCTURE) 和结构语法子句 (SYNTAX) 两部分组成,前者刻画该控制结构是如何由其他已知的函数和控制结构组合而成,后者则给出该控制结构的调用方式。函数定义用于完整地刻画软件功能分解的树形结构,通过使用语言提供的基本控制结构和用户定义的控制结构,逐层将描述软件功能的函数分解为子函数,形成一个树形结构,其叶结点函数可以是:语言定义的基本函数,用户用抽象数据类型定义机制刻画的操作,用函数定义机制定义的函数,用其他语言书写的函数,递归调用结点,尚未定义而需要在执行中模拟的函数等。

GSPEC 语言通过在函数分解树的叶结点中使用抽象数据类型中定义的操作,使语言的两种机制有机地结合起来,针对具体问题,用户可有所侧重。语言的图形化表示形象易读,便于软件的理解和维护,语言关于接口的精确刻画及关于抽象数据类型的正确性(终止性,一致性和完备性)验证设施有利于保证和验证软件规约的正确性。

GSPEC 语言已在 SUN 工作站和微型计算机上实现,并开发出若干具有实际意义的中小型软件,例如新构造找水系统等。

参考文献

徐家福等. 软件自动化. 北京: 清华大学出版社, 1994 (费宗铭)

Hopfield shenjing wangluo moxing

Hopfield 神经网络模型 (Hopfield neural network model) 一种单层全反馈的人工神经网络模型(后称之为 Hopfield 模型),它对推动人工神经网络研究的复苏起了很重要的作用。

Hopfield 对人工神经网络研究的贡献主要有:

(1) 把有反馈的神经网络看作一个非线性动力系统,提出了系统的全局 Lyapunov 函数(或称能量函数)的概念,用于系统稳定性的分析;

(2) 利用上述分析方法解决人工智能中的组合优化问题,如 TSP;

(3) 给出了利用模拟电子线路实现的连续 Hopfield 网络的电路模型,为进一步研究神经计算机创造了条件。

Hopfield 所提模型还表明可用作联想存储器。目前联想存储器与优化计算是 Hopfield 网络的主要应用方面。Hopfield 模型有两种,即离散的 Hopfield 网络和连续的 Hopfield 网络模型。

离散 Hopfield 神经网络是一离散时间系统,它可用一加权无向图表示,图中每个边都附有一权值,每个结点(代表一个神经元)都附有一阈值,网络的阶数等于图中的结点总数。设 N 是一 n 阶离散 Hopfield 神经网络,则 N 由 (W, Q) 唯一地定义,其中 W 是一 $n \times n$ 对称矩阵, w_{ij} 为连接结点 i 和 j 边上的权值; Q 是一 n 维矢量, Q_i 表示结点 i 的阈值。每个结点 i 取 1 或 -1 两个可能状态之一。以 $X_i(t)$ 表示 N 在 t 时刻结点 i 的状态,矢量 $X(t)$ 表示 N 在 t 时刻的状态,则 N 的每一结点 i 的下一状态由下列规则决定:

$$X_i(t+1) = \text{sgn}(H_i(t)) = \begin{cases} 1 & H_i(t) > 0 \\ -1 & \text{其他} \end{cases} \quad (1)$$

式中

$$H_i(t) = \sum_{j=1}^n w_{ij} X_j(t) - Q_i \quad (2)$$

由此可知:网络 N 的状态矢量 $X(t) \in \{1, -1\}^n$ 。式(1)称之为离散 Hopfield 神经网络模型。若 $w_{ij} = 0$ ($i=1, 2, \dots, n$),相应的模型称之为无自反馈的离散 Hopfield 神经网络,否则称之为自反馈离散

Hopfield 神经网络。

网络有两种工作方式:

(1) 串行(异步)方式 在任一时刻,只有某一神经元 i (随机地或按某种确定方式选择)按式(1)改变状态;

(2) 并行(同步)方式 对某一时刻,所有神经元同时按式(1)改变状态。(也可以一部分神经元同时改变状态,各部分以串行方式工作,称为部分并行方式。)

上述结论保证了神经网络并行计算的收敛性。

若 $N = (W, Q)$ 表示式(1)所描述的神经网络, W 为一对称矩阵,可以证明:

(1) 如果 N 工作在串行模式, W 的对角元素非负(包括对角元素为零的情况),则网络 N 总是收敛于稳定状态;

(2) 如果 N 工作在全并行模式,则网络 N 总是收敛于稳定状态或 Hamming 距离小于 2 的极限环。

上述结论保证了神经网络并行计算的收敛性。

连续 Hopfield 神经网络中,各个神经元状态取值是连续的,由于离散 Hopfield 神经网络中的神经元与生物神经元的主要差异是:①生物神经元的 I/O 关系是连续的;②生物神经元由于存在时延,因此其动力学行为必须由非线性微分方程来描述。为此,在 1984 年 J. J. Hopfield 提出了连续 Hopfield 神

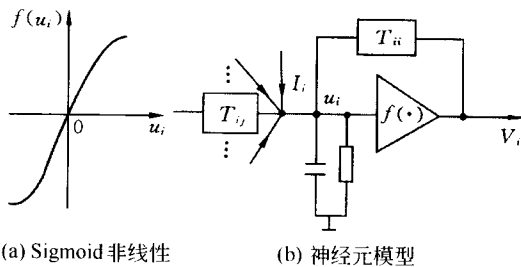


图1 连续 Hopfield 神经网络

经网络,它可用图1所示的电路实现,其动态方程可由下述微分方程式描述:

$$C_i \frac{du_i}{dt} = \sum_{j=1}^n T_{ij} V_j - \frac{u_i}{R_i} + I_i \quad (3)$$

$$V_i = f(u_i) \quad i = 1, 2, \dots, n$$

式中 $f(\cdot)$ 为连续可微的 Sigmoid 函数;

$$T_{ij} = T_{ji} \quad i, j = 1, 2, \dots, n$$

$$T_{ij} = 0 \quad j = i$$

$$\frac{1}{R_i} = Q_i + \sum_{j=1}^n T_{ij} \quad i = 1, 2, \dots, n$$

连续时间 Hopfield 神经网络式的计算能量函数定义为

$$E = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n T_{ij} V_i V_j + \sum_{i=1}^n \frac{1}{R_i} \int_0^{V_i} f_i^{-1}(x) dx - \sum_{i=1}^n I_i V_i \quad (4)$$

对于式(3),若 f^{-1} 为单调增且连续, $C_i > 0$, $T_{ji} = T_{ji}(i, j = 1, 2, \dots, n)$,则沿系统的运动轨道有

$$\frac{dE}{dt} \leq 0$$

当且仅当 $\frac{dV_i}{dt} = 0$ 时

$$\frac{dE}{dt} = 0 \quad i = 1, 2, \dots, n$$

式(3)的稳定平衡点就是能量函数 E [式(4)]的极小点,反之亦然。

同时,连续 Hopfield 神经网络式(3)以大规模非线性连续时间并行方式处理信息。网络的稳定平衡点对应于其计算能量函数 E 的极小点,网络的计算时间就是它到达稳定的时间,网络的计算在系统趋于稳态的过程中也就完成了。这也是式(3)用于神经计算及联想记忆的基本原理,也即神经计算机的基本原理。

参考文献

1. Hopfield J J. PNAS. USA, 1982, 79: 2554 ~ 2558
2. Hopfield J J. PNAS. USA, 1984, 81: 3088 ~ 3092
3. 焦李成. 神经网络系统理论. 西安: 西安电子科技大学出版社, 1990 (焦李成 阎平凡)

Internet

Internet(因特网) 全球最大的、开放式的、由众多网络互连而成的**计算机网络**。这是 Internet 的一般性定义,意味着全世界采用开放性协议的计算机都能互相通信。狭义的 Internet 指上述网中所有采用 IP 协议(参见**网际协议**)的网络互连而成的网络,通常把这样一个网称为**IP 因特网**。在 IP 因特网中, TCP/IP 协议的分组可通过**路由选择**相互传送。广义的 Internet 指 IP 因特网加上所有能通过路由选择至目的站的网络,包括使用诸如电子邮件这类应用层**网关**的网络,各种存储转发的网络以及采用非 IP 协议的网络。

Internet 是由美国的 ARPA 网发展和演化而成

的。ARPA 网是全世界第一个分组交换网。1969 年美国的国防高级研究计划署(DARPA)建立了一个只有 4 个结点的存储转发方式的分组交换**广域网**——ARPA 网,该网是为了验证远程分组交换网的可行性而进行的一项试验工程。

1972 年在首届国际计算机通信会议(ICC)上首次公开展示了 ARPA 网的远程分组交换技术。当时, ARPA 网已有约 20 个分组交换结点机(采用 BBN 公司开发的接口报文处理机 IMP)和 50 台主机。在总结最初的建网实践经验的基础上,开始了称为网络控制协议(NCP)的第二代网络协议的设计工作。随后 DARPA 又组织有关专家开发了第三代网络协议——TCP/IP 协议,并于 1983 年在 ARPA 网上正式启用,这是以后因特网得以迅速发展的关键因素之一。

1983 年 ARPA 网被分成两部分,一部分是专用于国防的 Milnet 网,剩下的部分仍称 ARPA 网。ARPA 网的建立,产生了网络互连的概念,即将各个独立的网络互连成一个更大的网络实体。在 1972 年的 ICC 会议上曾讨论过将世界上的研究网互连起来的问题。当 ARPA 网采用 TCP/IP 协议以后,上述想法变成了现实,使用称为**网关**的网络互连设备,形成了互连各种网络的网,称为**互联网**。其中以 ARPA 网为中心组成的新互联网称为 Internet。为区别于一般的互联网,Internet 的第一个英文字母用大写的 I。事实上,Internet 是多种技术发展的结果,包括将 ARPA 网、分组无线网、分组卫星网和**局域网**连接起来的技术,将各种网络互连的网络设备——网关的概念,将 IP 分组封装在更低层的网络分组内的方法以及 TCP/IP 协议等。其中网关的概念和 TCP/IP 协议是 Internet 的核心。从 1969 年 ARPA 网诞生到 1983 年 Internet 的形成是 Internet 发展的第一阶段,即研究试验阶段。

从 1983 年到 1994 年是 Internet 发展的第二阶段,其核心是 NSF 网的形成和发展,这是 Internet 在教育科研领域广泛使用的实用阶段。1986 年美国国家科学基金会(NSF)制定了一个使用超级计算机的计划,即在全美设置若干个超级计算机中心,并建设一个高速主干网,把这些中心的计算机连接起来,形成 NSF 网。NSF 网是一个 3 级分层的互联网,即 NSF 主干网、各个区域网和众多的校园网。1990 年到 1991 年,IBM 公司, MCI 公司和 Merit 公司共同组建了一个先进网络服务(ANS)公司,专门为 NSF 网提供服务。NSF 网的形成和发展,使它成为 Inter-

net 的最主要的组成部分。与此同时,很多国家相继建立自己的主干网,并接入 Internet,成为 Internet 的组成部分,如加拿大的 CA * net,欧洲的 EBONE 和 NORDUNET,英国的 PIPEX 和 JANET,以及日本的 WIDE 等。

Internet 最初的宗旨是用来支持教育和科研的活动,它不是营业性的商业组织,但是,随着 Internet 规模的扩大,应用服务的发展,以及全球化需求的增长,提出了一个新概念——Internet 商业化,并开始建立 AlterNet 网和 PSI 网(PSInet)这些商用 IP 网络。为了解决商用 IP 网络接入 Internet 的问题,1991 年宣布了一个解决方案,也就是采用称为商用 Internet 交换(CIX)互连点的结构,它由高速路由器和连接各 CIX 成员的链路组成,这些 CIX 的成员都是网络提供者,而不是网络用户。CIX 创造了更多的商业化机会,从此 Internet 就不仅是服务于教育、研究和政府部门的了。1994 年 NSF 宣布不再给 NSF 网运行维护经费支持,由 MCI 公司和 Sprint 公司运行维护,这样,不仅商业用户可以进入 Internet,而且 Internet 的经营也商业化了。

Internet 从研究试验的第一阶段到实用于科教的第二阶段,进而到商用的第三阶段的发展,反映了因特网技术和应用的成熟。

全世界已有几十万个网络、几千万台计算机接入 Internet,它拥有 2 亿多个用户,已成为全球最大的计算机互联网。更加重要的是,Internet 提供了极为丰富的信息资源和应用服务。它为发展信息网络技术和网络应用提供了丰富的经验,对信息市场的开拓和信息社会的发展具有深远的影响,已成为未来全球信息基础设施(GII)的原型。

常用的连接到 Internet 的方法有以下 3 种:

(1) 将计算机连接到一个局域网,这个局域网的服务器是 Internet 的一台主机;

(2) 利用串行线路网际协议(SLIP)或点对点协议(PPP),通过电话拨号方式进入 Internet 的某个主机;

(3) 通过电话拨号进入某个提供 Internet 服务的联机服务系统。

参考文献

1. Dem D P. The Internet Guide for New Users. New York: McGraw - Hill Inc., 1994

2. Comer D E. Internetworking with TCP/IP. New Jersey: Prentice Hall Inc., 1993 (胡道元)

Internet fuwu tigongzhe

Internet (因特网) 服务提供者 (Internet service provider, ISP)

为用户提供 Internet 接入服务和信息内容服务的公司。为选择一个可靠的、能快速连接的、强的客户支持的因特网服务提供者(ISP),对 ISP 的评估和选择是十分重要的。下面列举 8 个重要的评估准则。

(1) 是否满足服务对象的需求。

(2) 是否具备好的服务质量,包括可靠性、可用性和高性能。可靠性方面有 ISP 的主要交换设备和主干网是否有冗余,网络运行中心(NOC)是否有不间断供电系统等。可用性是指拨号用户拨号连通 ISP 的百分比,一般要求在 100 次拨号中最多有 5 次线路忙,即连通率为 95% 以上。高性能指的是 ISP 的主干网的传输速率以及用户连接的结点到主干网的速率高。

(3) 用户连接的结点是否靠近用户,ISP 是否直接连至国家或国际的 Internet 结点。

(4) 提供的服务类型是否满足用户需求。服务类型是指提供拨号服务和专有高速链路,提供的链路速率,以及提供的信息内容服务等。

(5) 提供的增值服务的情况,如是否根据客户需求提供安全选择、域名注册等服务。

(6) 对用户的支持程度,包括一天 24 小时是否连续运行,是否有专人为客户做咨询服务以及其他各种客户需要的服务。

(7) ISP 的经验多少,可从服务的历史、客户的情况等方面进行考察。

(8) 价格是否合理,即在满足上述要求的前提下是否有合理的价格,要注意不能单纯追求低价而不注重服务质量。

(胡道元)

Internet dizhi

Internet 地址 (Internet address) 用来标识 Internet 上的每台计算机或每个用户惟一确定它们位置的一种字符串。又称 IP 地址。

在 TCP/IP 协议中(参见 TCP/IP 协议集),规定分配给每台主机一个 32 位数作为该主机的 IP 地址。在 Internet 上发送的每个数据包都包含了一个 32 位的发送方地址和一个 32 位的接收方地址。

从概念上说,每个 IP 地址由两部分组成,即网络标识和主机标识。网络标识确定了该台主机所在

的物理网络,主机标识确定了在某一物理网络上的一台主机。

IP 地址的层次结构具有两个重要特性:①每台主机分配了一个惟一的地址;②网络标识号的分配必须全球统一,但主机标识号可由本地分配,无需全球一致。

将 32 位 IP 地址分成两部分,需要确定如何进行分配。网络标识部分需要足够的位数,从而保证能给 Internet 上每一个物理网络分配惟一的网路号。主机标识部分也需要足够的位数,以保证给物理网络上每台主机分配惟一的主机号。由于 Internet 上的网络的规模有很大区别,因此 IP 的编址方案将 IP 地址空间划分为 A、B、C 三种基本类,每类有不同长度的网络标识和主机标识,如图 1 所示。

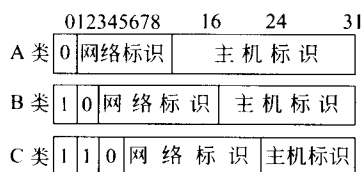


图 1 IP 地址编码

A 类地址分配给少数规模很大的网络,每个 A 类地址的网络有众多的主机。具体规定如下:32 位地址域中第一个 8 位为网络标识,其中第 0 位为 0,表示 A 类地址,其余 24 位均为主机标识,由该网的管理者自行分配。

B 类地址分配给中等规模的网络,每个 B 类地址的网络有较多的主机。具体规定如下:32 位地址域中前两个 8 位为网络标识,其中头两位为 10,表示 B 类地址,其余 16 位均为主机标识,由该网的管理者自行分配。

C 类地址分配给小规模的网络,每个 C 类地址的网络只有少量主机。具体规定如下:32 位地址域中前三个 8 位为网络标识,其中前三位为 110,表示 C 类地址,其余 8 位为主机标识,由该网的管理者自行分配。

IP 地址是 32 位二进制数,用户很难读数和输入,为了方便可用一种点分十进制表示法来表示。将 32 位数中每 8 位为一组,用十进制表示,利用小圆点分割各部分。最小值为 0,即一组内的所有位都为 0,最大值为 255,即组内所有位数都为 1。因此 32 位数用点分十进制表示的地址范围为 0.0.0.0 到 255.255.255.255。

根据上述规则,IP 地址的头 8 位,A 类为 0 到 127,B 类为 128 到 191,C 类为 192 到 223。还有两类不属于基本类的地址 D 类和 E 类。D 类用于广播传送至多个目的地址,头 4 位标识为 1110,因此 IP 地址的头 8 位范围为 224 到 239。E 类用于保留地址,头 4 位标识为 1111,因此 IP 地址的头 8 位范围为 240 到 255。

将一台计算机的 IP 地址映射到物理地址的过程称地址解析。一台主机或路由器在需要向同一物理网络上的另一台计算机发送数据时,先要进行地址解析。一台计算机无法解析远程网络上的计算机的地址。

将 IP 地址映射为物理地址的算法是不同的。它依赖于使用的协议和硬件编址方案。常用的地址解析算法有以下 3 种:①查表法 将地址映射关系放在内存中的一些表里,当解析地址时,通过查表得到解析结果;②相近形式计算法 通过简单的布尔和算术运算得出映射地址;③消息交换法 计算机通过网络交换信息得到映射地址。

TCP/IP 协议集包含一个地址解析协议(ARP)。ARP 定义了两类基本的消息:一类是请求消息,另一类是应答消息。一个请求消息包含一个 IP 地址和对相应物理地址的请求。一个应答消息包含发来的 IP 地址,也包含相应的物理地址。

ARP 规定了 ARP 消息如何在网上传递。一个 ARP 请求消息被放入一个帧后,被广播给网上所有计算机。每台计算机收到这个请求后,检测其中的 IP 地址,与 IP 地址匹配的计算机发送一个应答消息,而其他计算机则丢弃收到的请求,不发任何应答消息。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999
 2. Douglas E Comer. Internetworking with TCP/IP, Volume I, 3rd ed. Prentice Hall Inc., 1995
- (胡道元)

Internet jiben fuwu

Internet 基本服务(basic services of Internet) Internet 为用户提供的服务。它主要包括文件传送、远程登录和电子邮件服务。为提供这些服务,Internet 制定了相应的协议:

(1) 文件传送协议(FTP) 它是描述客户机与远程主机之间传送文件的协议(参见文件传送)。

用户和远程主机之间传送文件时,用户必须在远程主机上有合法的用户账号和口令以及相应的访问权限。

(2) 匿名文件传送协议(anonymous FTP) 它允许没有对方计算机注册账号和口令的 Internet 用户登录对方计算机,并获取所需的文件。Internet 有大量匿名 FTP 服务器,存放着数以百万计的各种共享文件(包括计算机软件、说明书、协议标准等)。在这种服务器上有一个名为 anonymous 的特殊注册账户,用户用这个名字去登录,用自己的电子邮件地址或本地服务器的提示作为口令去访问,便可以获得公开发布的文件。

(3) 远程登录协议(Telnet) 它是标准的 Internet 虚拟终端协议,实现客户机远程登录到 Internet 上的某台主机。用户登录到远程主机后,就成为该远程主机的虚拟终端用户,可共享该主机的软、硬件资源和数据库等。

(4) 简单邮件传输协议(SMTP) 它是描述客户机与远程主机之间传送电子邮件的协议(参见电子邮件)。(胡道元)

Internet tixi jiegou

Internet 体系结构(Internet architecture)

Internet 采用的协议及服务的概念性层次模型和结构。Internet 采用 TCP/IP 协议(参见 TCP/IP 协议集),故又称为 TCP/IP 互联网。如同开放系统互连基准(参考)模型,Internet 体系结构也是一种分层模型。它由基于硬件层次上的四个概念性层次构成,即应用层、传输层、IP 层和网络接口层。图 1 给出了这些概念性层次结构以及这些层次之间传送数据的形式。

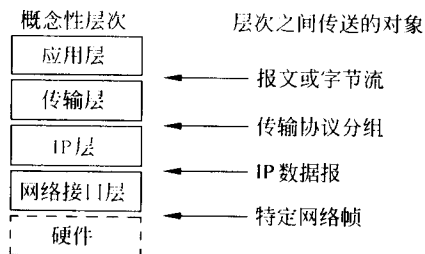


图 1 概念性层次结构

(1) 应用层 在最高层——应用层,用户调用应用程序来访问 TCP/IP 互联网络提供的多种服务。应用程序负责发送和接收数据。每个应用程序可选

择所需的传送服务类型,既可以传送独立的报文序列,也可以传送连续的字节流。应用程序的任务是将数据按要求的格式传送给传输层。

(2) 传输层 传输层的基本任务是提供发送端和接收端的应用层之间的通信,即端到端的通信。传输层管理信息流,提供可靠的传输服务,以确保数据无差错地、按序地到达。传输层软件将要传送的数据流划分成分组,并连同目的地址传送到下一层。

(3) IP 层 IP 层处理机器之间的通信。它接收来自传输层的请求,将带有目的地址的分组发送出去。IP 层将分组封装到数据报(参见网际协议)中,填入数据报头,使用路由算法以决定是直接数据报传送到目的主机还是传给路由器,然后把数据报传送到相应的网络接口来传送。IP 层还处理接收到的数据报,检验其正确性,并决定是由本地接收还是通过路由器传送到相应的目的站。

(4) 网络接口层 这是 Internet 体系结构的最底层。该层负责接收 IP 数据报并发送至选定的网络。网络接口包括一个设备驱动器,也可能是一个复杂的具有数据链路协议的子系统。

Internet 体系结构的概念性层次包含两个重要的分界线:一个是协议地址分界线,以区分高层和低层的寻址;另一个是操作系统分界线,以区分系统与应用程序。如图 2 所示。

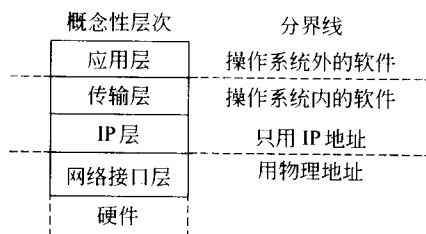


图 2 概念性层次的分界

高层寻址使用 IP 地址,低层寻址使用物理地址。应用程序 IP 层之上的所有协议软件只使用 IP 地址,而网络接口层处理物理地址。

通常将软件分成操作系统和非操作系统软件两部分。当协议软件集成到操作系统中后,在协议软件的底层之间进行数据传送的开销比应用层和传输层之间进行数据传送的开销要小得多。

从概念上讲,一个 TCP/IP 互联网提供三组服务。互联网服务的三个概念层及其相互的依赖关系,如图 3 所示。在最底层,无连接传送服务为其他

层的服务提供了基础。在第二层,一个可靠的传送服务为应用层提供了一个高层平台。最高层是应用服务层。

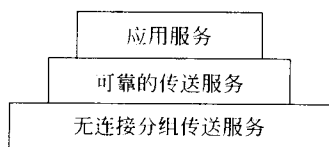


图3 互联网服务的三个概念层

TCP/IP的分层工作原理如图4所示,表示两台主机上的应用程序之间传输报文的路径。主机B上的第 n 层接收的正是主机A上的第 n 层发送出来的对象。在物理网络上传送的帧是相同的帧,以上各层收发分别为相同的数据报、相同的分组和相同的报文。

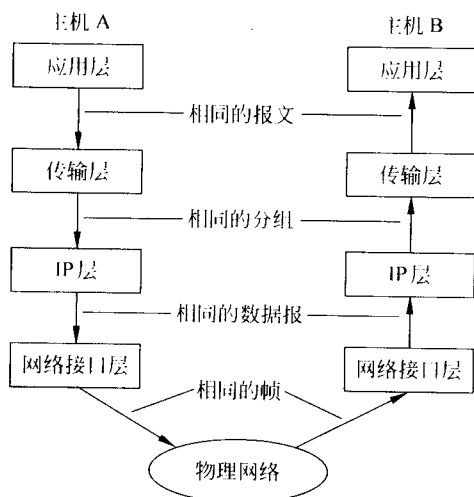


图4 TCP/IP 分层工作原理

Internet 软件是围绕着三个层次的概念化网络服务设计的。最基本的互联网服务由一个分组传送系统组成。该服务被定义为不可靠的、无连接的、尽最大努力传送的分组传送系统。所谓不可靠,指的是不能保证正确传送,分组可能丢失、重复、延迟或不按序传送,而且服务不检测这些情况,也不通知发送方和接收方。所谓无连接,指的是每个分组都是独立处理的,可能经过不同的路径,有的可能丢失,有的可能到达。所谓尽最大努力传送,指的是互联网软件尽最大努力来传送每个分组,只有当资源用尽或底层网络出现故障时,才会出现不可靠服务。

这种不可靠的、无连接传送的协议称为网际协议,简称IP协议。

参考文献

1. 胡道元. 计算机网络(高级). 北京:清华大学出版社,1999
2. Douglas E Comer. Internetworking with TCP/IP, Volume I, 3rd ed. Prentice Hall Inc., 1995

(胡道元)

Jackson xitong kaifa fangfa

Jackson 系统开发方法 (Jackson system development method) 一种面向数据结构的软件开发方法,该方法是以数据结构为基础,通过一组映射或转换过程来建立程序的结构。

JSD 是 Jackson System Development 的缩写。JSD 方法是由 M. A. 杰克逊提出的。他首先于 1972 年—1974 年间提出了 JSP,这是一种结构化程序设计方法,后来在 1978 年—1981 年间将其扩充为 JSD。

JSD 方法把系统开发分为描述和实现两个阶段。描述阶段建立一个与系统相关的客观世界的模型,并在此基础上确定系统功能。实现阶段在具体的软硬件环境下实现系统功能。

JSD 方法有 6 个步骤:

(1) 实体动作步骤:列出客观世界中与系统有关的原始实体和实体动作表,以及每一动作的属性表。这些表规定了系统功能的范围。

(2) 实体结构步骤:用结构图来表达每一个实体的诸动作间有序约束,得到一个结构图集。结构图中包括顺序、选择及重复等三种构造。

实体动作步骤和实体结构步骤的结果是一个客观世界的抽象描述。为了模拟客观世界,客观世界中的每一个实际进程需要有一个对应的模型进程。

(3) 初始模型步骤:说明模型进程是如何与其相应的实际进程进行联系的。JSD 方法支持数据流联结与状态向量联结两种方式,结果得到一个系统说明图。此外,软件开发人员还需写出模型进程的结构文本以便进一步说明模型进程。结构文本实质上是结构图的文本形式,它包含了从外界输入信息的操作,以及根据输入信息确定模型进程中操作执行的条件。

(4) 功能步骤:指明系统功能。指明的方式有 3 种:①以初始模型中的术语为基础,用自然语言按

规定的形式陈述功能。这种方式有利于软件开发人员与用户之间的通信。②在初始模型的系统说明图中加入功能规约。③用结构文本给出细化的功能规约。这3种方式一般要结合使用。

(5) 系统时序步骤:在时间上系统总是延迟于客观世界,而根据系统功能的需要,系统的不同部分在延迟的程度上总是有所区别的。为了完善系统规约,在本步骤中考虑这些延迟,并和系统用户一起讨论系统中各个不同部分的延迟程度。结果得到非形式的描述时序约束的文件。

以上5个步骤为描述阶段。

(6) 实现步骤:在具体的软硬件环境下实现系统功能。实现步骤的基本任务有3:①确定有多少真实或虚拟的处理器用于执行系统;②当处理器的数量小于进程的数量时,确定如何在进程间分配处理器;③在一个拥有多个进程的处理器上确定如何由多个进程共享处理器时间,即进行进程调度。

描述阶段得到的系统规约图展示了系统中有什么样的进程,进程之间如何联结以及系统边界的输入和输出,而结构图则展示了进程的结构,结构文本又进一步给出了进程的细化规约。实现阶段得到的调度进程指明了在具体环境下系统内部的事件发生的次序,系统实现图展示了系统的实现轮廓。在这些信息结构的指导下最后完成系统代码的编制。

JSD中的每一步骤的结果作为下一步骤工作的部分根据,但实际上开发并不是顺序地从上一步到下一步,回溯是不可避免的,JSD的目标是尽量减少这种情况的发生。

JSD方法是一种成熟的系统开发方法。它的基本特点体现在下面几个方面:①软件开发人员首先描述客观世界,即建立客观世界的模型,然后在模型的基础上提出系统功能。②动态的客观世界需要动态的模型来模拟。JSD中的模型是用具有动态概念的顺序进程来描述的,因此,JSD方法很适合于强调时序的应用。③JSD的实现技术允许软件开发人员在建立系统时自行决定进程的调度,而不是在系统运行时由操作系统来决定。④JSD方法致力于在开发过程中得到一些数据结构表示,如进程的结构图和结构文本,并以此为基础进行程序设计。

参考文献

Jackson M. A. System Development. London: Prentice-Hall International, 1983 (郝克刚 葛玮)

Java yuyan

Java 语言 (Java language) 一种简捷的、面向对象的、用于网络环境的程序设计语言。Java语言是由 SUN MicroSystem 公司于 1995 年 5 月正式对外发布的。

Java 语言的基本特征是:简捷易学、面向对象、适用于网络分布环境、解释执行和多线程、具有一定的安全健壮性。

简捷易学——最初开发 Java 语言的本意是为家用电器的程序控制而用的。它坚持面向对象的基本原理,但又避免了运算符重载、多重继承等复杂概念。它基本上是一种解释执行的语言,而且其基本解释程序和对于类的支持只有 40 kB 左右,加上标准类库和线程支持也只有 215 kB 左右,因此系统开销较小,适用于小型的信息处理和信息环境。它由于能够实现自动废区收集,因此也简化了程序设计的内存管理工作。

面向对象——在坚持面向对象方法的基础上,Java 提供了极简单的类机制,以及很有效的接口模型。Java 的对象中封装了其状态变量和相应方法,实现了模块化和信息隐蔽;而通过类的继承机制,子类可以使用父类所提供的方法,从而实现了代码复用。

适用于网络分布环境——Java 是面向网络应用的语言,通过它所提供的类库,可以处理 TCP/IP 协议规程,可以通过 URL 地址在网络上访问其他对象,能较方便地与其他计算结点协同工作。

解释执行和多线程——Java 解释程序能直接对 Java 的字节码进行解释执行,由于可从字节码获得部分编译信息,因此使得连接过程更加简捷。Java 所提供的多线程机制可使应用程序并行执行,其同步机制也有助于实现数据共享。

安全健壮——由于 Java 提供了自动废区收集、面向对象的异常处理、自动捕获类型声明中的常见错误、一切对内存的访问都必须通过对象的实例变量实现(不支持指针)等手段,因此 Java 可防止部分故障,具有一定的安全健壮性。

由于 Java 具有以上特性,所以已受到各种应用领域的重视,取得很快的发展,在因特网上已推出了用 Java 语言编写的多种应用程序。但 Java 还不很成熟,尚有许多可改进之处。随着 Java 芯片、Java OS、Java 解释执行和编译、Java 虚拟机技术的日趋先进,Java 语言将更加完善,发挥更大的作用。

参考文献

1. The Java Language: An Overview. From <http://java.sun.com>
2. 王克宏主编, 郁欣等. Java 语言编程技术. 北京: 清华大学出版社, 1997
3. 王克宏主编, 李京华等. Java 虚拟机规范. 北京: 清华大学出版社, 1996 (汪成为 王克宏)

Kerberos jianbie

Kerberos 鉴别 (Kerberos authentication)

一种使用对称密钥加密算法来实现通过可信第三方密钥分发中心 (KDC) 的身份认证系统 (参见私钥密码技术、鉴别)。它是由美国麻省理工学院 (MIT) 为了保护 Athena 项目中的网络服务和资源而开发的。Kerberos 第 5 版的协议已被 Internet 工程任务部 IETF 正式接受为征求意见稿 RFC 1510。

Kerberos 在学术界和工业界都获得了广泛的支持, 被众多系统选作身份认证的基础平台。例如开放软件基金会 (OSF) 开发的分布式计算环境 (DCE) 就是以 Kerberos 为身份认证平台的。在国外应用最广泛的分布式文件系统 (AFS) 也采用了 Kerberos 作为身份认证平台。目前各主要操作系统都支持 Kerberos 认证系统。例如, SUN Microsoft 公司在其高端服务器产品 Windows NT 5.0 中也支持 Kerberos 系统。Kerberos 实际上已经成为工业界的事实工业标准。

Kerberos 使用对称密钥加密算法来实现通过可信第三方——密钥分发中心 (KDC) 的认证服务。它提供了网络通信方之间相互的身份认证手段, 而且不依赖于主机操作系统和地址。Kerberos 设计的目标是在开放网络上运行, 不要求网络上所有主机的物理安全, 同时还假设通过网络传输的包可以被任意截获、修改和插入。Kerberos 系统非常适合在一个物理网络并不安全的环境下使用。它的安全性已经过了实践的考验。

Kerberos 协议中有三个通信参与方, 即需要验证身份的通信双方再加上一个双方都信任的第三方 KDC。当某个网络应用进程需要访问另外一个服务进程时 (例如向远程文件传输 (FTP) 服务器发起 FTP 连接), 它首先需要向 FTP 服务器证实自己的身份, 同时也要确认该 FTP 服务器的身份, 这样就构成了双向的身份认证。发起认证服务的一方被称为客户方, 客户方需要访问的对象被称为服务器方。在 Kerberos 中客户方是通过向服务器方递交自己的

“凭据”来证明自己的身份的。该凭据是由密钥分发中心 (KDC) 专门为客户方和服务器方在某一阶段内通信而生成的。凭据中包括客户方和服务器方的身份信息、在下一阶段双方使用的临时加密密钥 (称为会话密钥), 还有证明客户方拥有会话密钥的身份认证者信息。身份认证者信息的作用是防止攻击者在将来将同样的凭据再次使用。

Kerberos 在 Needham - Schroeder 原始模型中加入了时间标记以检测是否受到重放攻击。时间标记与 KDC 共享的密钥共同构成了客户方或者服务器方相信它接收到的凭据的真实性的基础。为了提高安全性能, 一个 Kerberos 凭据只在一段有限的时间内有效, 称为凭据的生存期。当生存期过后凭据自动失效, 以后的通信必须从 KDC 获得新的凭据进行认证。

Kerberos 保持一个客户方以及密钥的数据库, 这些密钥仅在 KDC 与客户方之间共享, 不能被第三方知道。如果客户是用户, 那么该密钥就是用户口令经过散列算法生成的; 如果其他网络服务需要使用 Kerberos 认证服务, 则需要进行登记并且在登记时协商共享密钥, 而这些密钥往往是机器随机生成的。

参考文献

- Kohl J, Neuman C. The Kerberos Network Authentication Service (V5). RFC 1510, 1993 (胡道元)

Larch yuyan

Larch 语言 (Larch language) 代数方法和一阶谓词方法相结合的形式软件规约语言簇, 它旨在开发出各种技术和工具以支持形式规约的有效使用。Larch 语言由美国麻省理工学院 J. V. Guttag 等人于 1983 年提出。

Larch 形式软件规约语言簇的主要目的是为各类程序设计语言的程序模块提供必要的描述手段, 考虑到不同的程序设计语言常有不同的语法表示和语义定义, 因而程序模块的规约就与某一特定的程序设计语言有关, 为使和程序设计语言有关的特性局部化, Larch 提供了描述各类程序设计语言的程序模块共性的共享语言和描述各个特定程序设计语言个性的接口语言。Larch 语言簇中每一成员由共享语言及某一特定接口语言组成。

Larch 共享语言采用代数方法, 其描述单位为 trait, 提供一组供接口规约使用的类符和操作, 由语法部分和公理部分组成。语法部分刻画了操作符的

型构,公理部分以等式的形式给出。每一 trait 定义了一个相关的理论,作为它的语义,区别于常规的以模型作为语义的处理。trait 定义中通过使用 imports 和 includes 子句以便从较小的 trait 组合成较大的 trait,通过定义生成操作集和识别操作集扩展所定义的理论。Larch 共享语言的重要特征之一是引入多种语言成分以便于语义检查,trait 中提供了 implies 子句用于表达用户希望相应 trait 的理论所蕴涵的定理。

接口语言的目的是从软件规约的角度,将共享语言和特定的程序设计语言有机结合在一起,并提供描述程序模块中相应运算的机制。其中与特定语言有关的成分,如过程名、形式参数、类型、异常处理等描述的语法借助于该语言的语法,而其中过程的功能刻画用基于一阶谓词的前后断言的形式,描述中可以使用共享语言中 trait 的操作和类型。

Larch 语言通过区分共享语言和接口语言,使和程序设计语言有关的特性局部化,其共享语言具有易组合和强调语义检查的特点。共享语言采用代数方法,接口语言基于一阶谓词演算,具有较好的理论基础。

Larch 已基于项重写系统实现,可用于分析其形式规约,检查其中的语义约束。Larch 语言已用于描述诸如图书馆系统等一系列软件规约。

参考文献

Guttaj J V, Horning J J. Report on the Larch Shared Language. Science of Computer Programming, 1986, (6): 103 ~ 134 (费宗铭)

LINPACK jizhun chengxu

LINPACK 基准程序 (LINPACK benchmark) 一套用以评测计算机计算性能的通用数学程序集,其功能偏重于对系统在解算基本线性方程方面性能的测试。该程序集由美国田纳西大学计算机科学部的 Jack J. Dongarra 开发完成,最初发表于 1976 年。目前 LINPACK 基准程序已成为工业上广泛用来评测不同计算机系统,从超级计算机到个人计算机计算性能的行业标准。

LINPACK 程序用 FORTRAN 语言写成,特别适合于科学与工程计算等线性方程密集型处理的性能评测。由于这类问题存在着大量的浮点运算,所以适于用它来评测计算机的浮点性能。LINPACK FORTRAN 程序的绝大部分时间都用于频繁调用包

含许多浮点操作的一组基本线性代数子程序 BLAS。为了显示产品的系统效能,有些计算机厂商用汇编语言码 BLAS 来替代标准的 FORTRAN BLAS,因为 FORTRAN 编译器产生的代码不如汇编语言代码优化,所以它们常用汇编代码 BLAS 作为其 LINPACK 的测试标准。

用优化的 LINPACK 基准程序和扩大的矩阵(阶数 $n = 1\,000$)可以测得系统接近于理论峰值性能指标 TPP。

所谓理论峰值性能,是系统处理器数量和时钟周期计算所得的理论结果,是机器的最高极限速度。

LINPACK 测试程序可以有单精度和双精度两种状态,其运行结果以百万[次]浮点运算每秒 MFLOPS 来表示。

采用 LINPACK 基准程序当矩阵阶 $n = 100$ 时,所测得的系统计算性能值常仅为该系统理论峰值的 10% 以下,但其 TPP 值一般可达理论峰值的 70% 以上。由于 LINPACK 测试运行的是实际使用程序,比理论峰值更能反映系统的真实计算性能。

(詹文岛)

Linux caozuo xitong

Linux 操作系统 (Linux operating system)

一种和国际上流行的 UNIX 同类的作为自由软件的操作系统。UNIX 是商品软件,而 Linux 则是一种自由软件。它是遵循 GNU(该词最初为自由软件倡导者 Richard Stallman 为其操作系统所起之名)组织倡导的通用公共许可证(GPL)规则而开发的,其源代码可以免费向一般公众提供。因此,Linux 多年来在教育界和学术界十分流行,成为广大计算机编程人员学习研究的对象。许多人还对之进行修改补充,以满足特定的需要。

Linux 是由芬兰科学家 Linus Torvalds 于 1991 年编写的一个操作系统内核。当时他还是芬兰赫尔辛基大学计算机系的学生,在学习操作系统课程中,自己动手编写了一个操作系统原型,从此,诞生了一个新的操作系统。Linux 把这个系统放在 Internet 上,允许自由下载,许多人对这个系统进行改进、扩充、完善,并做出了关键性的贡献。Linux 由最初一个人写的原型演化成在 Internet 上由无数志同道合的程序高手参与的一场活动。

Linux 属于自由软件,而操作系统内核是所有其他软件最为基础的支撑环境,再加上 Linux 的出现时间正好是 GNU 工程已完成了大部分操作系统外

围软件,水到渠成,可以说 Linux 为 GNU 工程画上了一个圆满句号。除了 Linux 外,还有许多自由软件,如 NetBSD、OpenBSD 等都是较优秀的具有版权的 Unix 类操作系统。Apache 也是一个著名的自由软件,已在服务器上广泛使用,支持包括 Linux, Free BSD, Solaris 及 HP-UX 等很多操作系统平台。

短短几年, Linux 操作系统已得到广泛使用。1998 年, Linux 已在构建 Internet 服务器上超越 Windows NT。计算机的许多大公司如 IBM, Intel, Oracle, Sun 等都大力支持 Linux 操作系统。各种成名软件纷纷移植到 Linux 平台上,运行在 Linux 下的应用软件也越来越多。Linux 的中文版已开发出来,并开始在中国流行。同时,也为发展我国自主操作系统提供了良好条件。

Linux 是一个开放源代码、Unix 类的操作系统。它除继承了历史悠久和技术成熟的 Unix 操作系统的特点和优点外,还进行了许多改进,成为一个真正的多用户、多任务通用操作系统。1993 年,第一个产品版 Linux 1.0 问世时,全部按自由扩散版权进行扩散,即公开源码,不准获利。不久发现这种纯粹理想化的自由软件会阻碍 Linux 的扩散和发展,特别限制了商业公司参与并提供技术支持的积极性。于是 Linux 转向通用公共许可证 (GPL) 版权,除允许持有自由软件的各项许可权外,还允许用户出售自由软件拷贝程序。这一版权上的转变后来证明对 Linux 的进一步发展十分重要。

从某种意义上来说, Linux 是 Unix 和 Internet 国际互联网结合的产物。自由软件 Linux 是一个充满生机、已有巨大用户群和广泛应用领域的操作系统,它是惟一能与 UNIX 和 Windows 较量和抗衡的一种操作系统。

从技术上讲, Linux 具有如下特点: ①继承了 UNIX 的优点,又有许多改进,能紧跟技术发展潮流,具有很强的生命力; ②通用的操作系统,可作为 Internet 上的服务器,可用作网关路由器,可用作文件和打印服务器,也可供个人使用; ③内置通信联网功能,可让异种机联网; ④开放的源代码,有利于发展各种特色的操作系统; ⑤符合 POSIX 标准,各种 Unix 应用可方便地移植到 Linux 下; ⑥提供庞大的管理功能和远程管理功能; ⑦支持大量外围设备; ⑧支持 32 种文件系统,如 Ext 2、Ext、Xiafs、Isofs、Hpfs、MSDOS、UMSDOS、Proc、NFS、SYSV、Minix、SMB、Ufs、Ncp、VFAT、AFFS 等; ⑨提供图形用户接口 (GUI),有图形接口 X-Windows,有多种窗口

管理器; ⑩支持并行处理和实时处理,能充分发挥硬件性能; ⑪可自由获取源代码,在 Linux 平台上以低成本开发软件。

参考文献

1. <http://freesoft.cei.gov.cn>
 2. <http://www.linux.org>
 3. 汤荷美,董渊,李莉等. Linux 基础教程(1) 操作系统基础. 北京:清华大学出版社,2001
- (周锡令 费翔林)

LISP yuyan

LISP 语言 (LISP language) 一种表处理语言。LISP 是 list processing 的缩略语,它是由 John McCarthy 于 20 世纪 50 年代后期提出的,其理论基础是 λ 演算。LISP 语言的程序由一些函数组成。函数的构造和数学上递归函数的构造方法类似,即从几个基本函数出发,通过一定的方法构造出新的函数。在 LISP 中常用的关于表处理的函数有 cons, car, cdr, null 等。下面的 LISP 程序定义了一个计算阶乘的函数 fac:

```
(define fac (n)
  (if (= n 0)
    1
    (* n (fac (- n 1)))))
```

LISP 语言的主要特点有:

- (1) 使用条件表达式书写递归函数;
- (2) 使用表数据结构和高阶函数进行编程;
- (3) 使用 S-表达式表示数据结构;
- (4) 在实现上使用 cons 结构和无用单元回收策略进行存储管理。

LISP 是最早的函数式程序设计语言之一。它有许多不同的实现和扩充,典型的有 MACLISP、InterLISP、EuLISP 等,其中最有影响的是 Common LISP,另外,有与面向对象技术结合的 CLOS 语言,并行语言 StarLISP,以及称为现代 LISP 的 Scheme 等。

LISP 已广泛用于人工智能研究的各个领域。

参考文献

1. McCarthy J. Recursive Functions of Symbolic Expressions and their Computation by Machine. CACM, 1960, 3(4)
 2. Steel Jr G L. Common Lisp the Language. Dittal Press, 1984
- (黄林鹏 孙永强)

Listserv luntan

Listserv 论坛 (Listserv) 因特网上提供的,方便用户之间交流信息的一种电子论坛。电子论坛是利用电子邮件的存储转发功能来实现的。在因特网上有各种各样的兴趣讨论组,用户加入自己感兴趣的讨论组后,可发送信件给论坛电子邮件服务器,服务器处理后,该信件就自动送一份拷贝到兴趣讨论组的每个用户的邮箱中。

Listserv 论坛的特点是用户可以及时获得讨论组成员发布的信息,进行交流;用户可以自由选择加入或离开讨论组,但不能控制讨论组发送来的信息。为了适当地控制信息的发布,电子论坛有一种变通形式,称为**电子杂志**。它配备有编辑人员,对搜集到的信息,经过筛选和编辑后分发给订阅用户。电子杂志一般是定期的。(胡道元)

LR(k) wenfa

LR(k) 文法 (LR(k) Grammar) 通过查看位于当前输入位置右边的 k 个输入符号,就能完成分析的文法。这里,LR 指从左向右, k 代表查看 k 个输入符号。LR(k) 文法,是一类上下文无关文法,我们可为它构造无回溯“有效”的分析器。所谓“有效”是指在处理输入长度为 n 的字符串的过程中,能够在 $C_1 n$ 时间与 $C_2 n$ 空间内完成分析,其中, C_1 , C_2 为两个常数。

一般地说,大多数用上下文无关文法描述的程序语言都可用 LR 分析器予以识别。LR 分析法比算符优先分析法或其他“移进-归约”技术应用得更加广泛,而且识别效率甚佳。比普通的无回溯的自上而下方法也要好。LR 分析法在从左至右扫描输入串时,就能发现其中的语法错误,并能准确地指出出错位置。

这种分析法的一个主要缺点是,若用手工构造分析程序则工作量相当大。因此,必须求助于自动产生这种分析程序的产生器。这种产生器称为 LR 分析程序自动产生器。应用这种产生器,能自动产生一大类上下文无关文法的 LR 分析程序。这种产生器还能指出文法有歧义的情形或难于分析的特殊结构。

构造 LR 分析器的主要任务是构造分析表。分析表恰好识别文法所产生的全部语句。分析表的构造通常有四种方法。第一种,也是最简单的一种,叫做 LR(0) 表构造法。这种方法的局限性极大,但它

是建立其他较一般的 LR 分析法的基础。第二种叫做简单 LR(简称 SLR) 表构造法。虽然,有一些文法构造不出 SLR 分析表,但是这种构造法是一种容易实现又极有使用价值的方法。第三种叫做规范 LR 表构造法,这种分析表能力最强,适用一大类文法,但实现代价过高,分析表的体积太大。第四种叫做向前 LR 表构造法(简称 LALR)。这种分析表的能力介于 SLR 和规范 LR 之间,稍加努力,就可以高效地实现。

LR 分析法由 D. E. Knuth 于 1965 年首先提出,1969 年,De Remer 把这个方法简化成实际可用的算法,即 SLR 和 LALR 算法。给定 LR(1) 系统,优化过程实质上将导出 LALR 分析器。同时,LR(0) 文法也可转化为 SLR 系统。但并不是所有 LR 文法,这种优化均可以做到。因此,以这种方式优化的语言集比用 LR 文法系统表示的语言集小得多。但对多数程序语言来说,用 SLR 分析也就足够了。

参考文献

陈火旺,钱家骅,孙永强. 程序设计语言编译原理(第二版). 北京:国防工业出版社,1984

(陈火旺 贵可荣)

Miranda yuyan

Miranda 语言 (Miranda language) 一种非严格的纯函数式程序设计语言。它是 1985 年至 1986 年间由 D. Turner 设计的。Miranda 的程序为一组递归定义的函数和其他数据对象。Miranda 使用卫士式等式或模式匹配表示情况分析。一个基于卫士式等式的阶乘函数可写成

$$\begin{aligned} \text{fac } x &= 1 & x &= 0 \\ &= x * \text{fac}(x-1) & \text{otherwise} \end{aligned}$$

而一个基于模式匹配的阶乘函数可表示为

$$\begin{aligned} \text{fac } 0 &= 1 \\ \text{fac}(x+1) &= (x+1) * \text{fac}(x) \end{aligned}$$

在实现时,前者的控制特征是显式的,而后者是隐式的。Miranda 是第一个被广泛传播的具有惰性计值语义和多态强类型的函数式程序设计语言。它已运行于世界上 600 多个场所,包括在 250 所左右的大学用于教学。Miranda 系统工作于 UNIX 环境下,是 Research Software Limited 公司的商业产品。

[注] Miranda 是 Research Software Limited 公司的注册商标。

参考文献

1. Turner D. Miranda: A Non - Strict Functional Language with Polymorphic Types. Proceedings FPCA' 85, Springer LNCS, 201: 1 ~ 16
2. Turner D. An Overview of Miranda. SIGPLAN Notices, 1986, 12(12): 158 ~ 166 (黄林鹏 孙永强)

ML yuyan

ML 语言 (ML language) 一种严格的函数式程序设计语言。ML 是 metalanguage 的缩略语,它是 20 世纪 70 年代中期由 R. Milner 提出的,当时的目的是作为由 M. J. Gordon 等人开发的称为 LCF 系统的命令语言,该系统用于证明称为 $PP\lambda$ 的程序理论的性质。ML 是一种强多态类型的语言,一个 ML 程序是一个包含变量定义和函数作用的表达式序列。ML 虽被称为是函数式语言,但它具有引用(即变量可以赋值)的概念并且它的 I/O 系统也引入了副作用。

1984 年,一个由爱丁堡大学、剑桥大学、贝尔实验室和 INRIA 组成的委员会在 R. Milner 的领导下对 ML 语言进行标准化工作,最后的结果加进由 D. MacQueen 设计的模块机制并被称为 SML(Standard ML)。SML 是一个比较完善和实用的函数式程序设计语言,它有高阶函数功能、I/O 机制、参数化的模块系统和完善的类型系统。一个简单的例子是:

```
let fun sum i tot = if i = 0 then tot
    else sum(i - 1) (tot + i)
in sum 10 0
end
```

它计算 $1 + 2 + 3 + \dots + 10$ 的值。

ML 语言的其他变形有 Lazy ML, CAML, CAML-Light 等。

参考文献

1. Milner R, et al. The Definition of Standard ML. MIT, 1990
2. Milner R, Tofte M. Commentary on Standard ML. MIT, 1991 (黄林鹏 孙永强)

Modula-2 yuyan

Modula-2 语言 (Modula-2 language) 一种程序设计语言。它直接源于 Modula,二者均由 N. Wirth 于 20 世纪 80 年代初设计,它们继承了 PAS-

CAL 语言中良好的传统构造,其中包括典型的控制结构、数据类型和过程等概念。并弥补了 PASCAL 语言的不足,增加新的设施,即引进了模块和进程概念,增加低级设施,采用更为系统化的语法等。

其主要特点是:

(1) 模块性 在 Modula-2 中,把模块分成两个语法成分,即定义性模块和实现性模块,并把那些在模块外可见的对象列在一个明显的移出表中,在模块内引用模块外的对象列在移入表中。Modula-2 的模块设施有利于实现模块的分别编译。

(2) 良好的控制结构 每个控制结构都以关键字结尾,可避免歧义性,减少不必要的“BEGIN”。提供了丰富的循环控制结构。特别引进 LOOP 和 EXIT 语句,可以较方便地代替 GOTO 语句。

(3) 输入输出功能由一组模块来实现,这组模块的层次结构反映了输入输出功能的多种抽象级别。

(4) 与机器和实现有关的低级设施放在伪模块 SYSTEM 中,可在高级语言级上实现。

(5) 提供一个层次较高的模块 Processes 和协同程序,有利于实现并发处理。

目前,世界上已经开发了近百个 Modula-2 编译系统。欧洲、加拿大、澳大利亚等不少大学已经用 Modula-2 代替 PASCAL 语言作为计算机科学系本科生的第一门程序设计课。

1984 年英国标准化学会开始进行 Modula-2 标准化工作,国际标准化工作始于 1987 年(ISO/IEC JIC1/SC 22 WG 13),标准化的一个新颖和重要的方面是首次采用形式化方法,用形式化定义(VDM-SL 和扩充的 BNF)来表达语言各成分的语法和语义。并伴以自然语言规约语句和注释。

参考文献

1. Wirth N. Programming in Modula-2. 2nd Edition. New York: Springer-Verlag, 1983
2. 郑国梁. Modula-2: PASCAL 的直接后继语言. 微型计算机 1987, (3) (郑国梁)

NP lei wenti

NP 类问题 (class NP of problems) 计算复杂性理论中一类重要问题,即多项式时间可验证的问题。

在计算复杂性理论中,问题由无数具体的实例组成,每个实例经编码可表示成一个字符串。所有具有肯定回答的实例所对应的字符串组成一个语言

(参见 P 类问题)。

图灵机由一个有限控制器,一条输入带和相应的读写头组成。图灵机根据有限控制器的状态和读写头所读到的字符,将执行以下三个操作:读写头在其所在方格内写下一个字符,改变机器状态,最后读写头向左(或向右)移动一格。如果对于任意一个状态和一个字符,要执行的动作(包括上述三个操作)都是惟一的,这种图灵机叫做**确定型图灵机**。如果要执行的动作有无穷多个可供选择,那么此时从第一步到最后一步,图灵机就有许多种可能的动作序列。如果规定,对给定的输入字符串 w ,所有这些动作序列中只要有一种导致接受状态,机器就接受 w ,这种图灵机就称为**非确定型图灵机**。

一个语言 L 称为是属于语言复杂性类 NP,如果存在一个非确定型图灵机 M 和一个多项式 p ,对于每个长度为 n 的字符串 w ,且 w 是 L 的字符串, M 都能在 $p(n)$ 步内停机,并接受 w 。

一个问题称为是 NP 类问题,如果它对应的语言属于 NP 类。从定义可以看出,P 类问题都是 NP 类问题,且 NP 类问题似乎要比 P 类问题更广。实际上,假定非确定型图灵机在每一步,都恰有两个动作可供选择。此时,要确定一个字符串是否被接受(即问题是否有肯定回答),图灵机在 $p(n)$ 步内,要考察 $2^{p(n)}$ 种不同的序列(这需要指数时间!)。现实中,许多问题正是以这种方式判断一个实例是否有肯定回答的,它们都是 NP 类问题,看上去很难有多项式时间算法。但 NP 是否真比 P 大,至今尚无定论。

非确定型图灵机还有另一种模型(见图 1)。它由一个确定型图灵机和一个猜测模块组成。对于给定输入字符串 w ,首先由猜测模块在带上-1 格及其左方任意写下(猜测)一个字符串,然后转由有限控制器控制,按确定型图灵机方式工作。由于猜测的任意性,机器可能有许多种动作序列,只要其中一种导致接受状态,机器就接受 w 。已经证明,以上两种非确定型图灵机模型彼此等价。正因为这个模型,NP 类问题被看成多项式时间可验证问题。

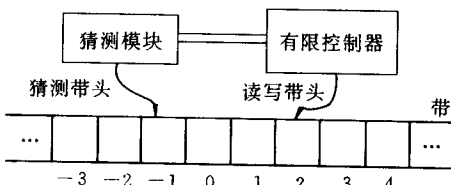


图 1 非确定型图灵机的另一种模型

参考文献

Garey M R, Johnson D S. Computers and Intrac-tability. San Francisco: Freeman, 1979 (徐美瑞)

NP wanquan wenti

NP 完全问题 (NP complete problem) 计算复杂性理论中的一类重要问题,其中每个问题都是 NP 类问题,但是否任何一个 NP 完全问题都是 P 类问题,目前尚无定论。NP 完全问题的研究,在理论上和实践中都有重要意义。

假定给了两个问题 q 和 q_0 ,如果存在一个确定型图灵机 M 和一个多项式 p ,对于 q 中任意一个长度为 n 的问题实例 x , M 都能在 $p(n)$ 时间内,计算出 q_0 的一个实例 y ,使得 x 是 q 中有肯定回答的实例,当且仅当 y 是 q_0 中有肯定回答的实例,就称问题 q 多项式时间多一归约到问题 q_0 。显然,如果有这样的归约, q 的求解问题就归结为 q_0 求解的问题。

对于一个问题 q ,如果 q 属于 NP 类,且 NP 中任意一个问题都可以多项式时间多一归约到 q ,就称 q 为 NP 完全问题。在现实生活中存在大量 NP 完全问题,它们分布在计算机科学、数学、逻辑学和运筹学等许多学科领域,总数已达数千。下面是几个典型的 NP 完全问题。

(1) **旅行商问题**: 给定 n 个城镇和一个界限 B ,以及城镇间的距离 $d_{i,j}$ ($1 \leq i, j \leq n$),问是否有一条旅行路线,经过每个城镇一次且仅一次,最后返回出发城镇,且其总路程不超过 B 。

(2) **划分问题**: 给定 n 个自然数,是否能将它们分成两部分,使得两部分自然数各自的和数彼此相等。

(3) **可满足性问题**: 对于任意给定的由“与”,“或”,“非”和逻辑变元组成的布尔表达式,是否有对式中各变元的一组赋值,使该表达式的值为真。

(4) **带优先次序的调度问题**: 有 m 个处理机和一个任务集合,每个任务所需时间为 1,已知任务间的优先次序(任务间不一定都有次序)和一个截止时间 D 。问是否有一个 m 个处理机的调度方法,满足优先次序要求,且在截止时间 D 以前完成全部任务。

由于 NP 类中任意一个问题都可以多项式多一归约到 NP 完全问题,因此,直观上人们可以认为 NP 完全问题是 NP 类中“最难”的问题。对于 NP 完全问题,一般都有一个明显的指数时间算法,但按目前研究结果,要找到一个现实可计算的算法,即多项

式时间算法,却十分困难,甚至很可能根本不存在这样的算法(参见 NP 完全性理论)。NP 完全问题的求解,只有另辟蹊径,例如寻找多项式时间近似解法。

参考文献

Garey M R, Johnson D S 著. 计算机和难解性: NP 完全性理论导引. 张立昂, 沈泓, 毕源章译. 北京: 科学出版社, 1987 (徐美瑞)

NP wanquan wenti jinsi fangfa

NP 完全问题近似方法 (approximate method for NP complete problem) 近似求解 NP 完全问题的各种方法。十几年前,人们着手研究 NP 近似方法,并已为若干 NP 完全问题找到了近似快速求解算法。这里的“近似”有三种不同的含义。第一,算法并不是对于问题的全体实例都算得好,它对少数实例的计算可能不好。第二,对 NP 难度的组合优化问题,算法对全体实例的计算都是快的,但计算出的函数值含有一定的误差。第三,算法对每个实例都依很接近于 1 的概率算得又快又精确。近十年的经验证明,按第一、第三种近似意义可能找到实际效率很高的求解算法,而按第二种近似意义虽然可能找到理论上漂亮的算法但它们的实际效率都会很低。

NP 完全问题近似方法的理论研究大体上沿着以下 4 个方向发展: (1) 对具体的 NP 完全问题设计出具有多项式时间复杂度的近似求解算法,并估计其解的近似效能; (2) 对某些 NP 完全问题证明根本就不存在近似性能不是太坏的完整的快算法,除非 $P = NP$; (3) 对具体的 NP 完全问题,尽力找出其具有多项式时间复杂度求解算法的最大子问题类; (4) 在高维空间中,引进蒙特卡洛法,牺牲绝对的必然性,换取组合爆炸现象的消失,为 NP 完全问题得出实际上效能很高的快速求解算法。

参考文献

1. 黄文奇. 求解 Covering 问题的拟物方法——NP 难度问题的一个处理途径. 计算机学报, 1989 (8)

2. Huang Wenqi and Moss Sweedler. A Practical, Feasible Square Packing Algorithm for Chip Manufacture in VLSI. Technical Report 1992-20, Mathematical Sciences Institute, Cornell University, U. S. A.

3. Hougardy S, Prömel H J, Steger A. Probabilistically Checkable Proofs and their Consequences for

Approximation Algorithms. Forschungsinstitut für Diskrete Mathematik, Universität Bonn, Nassestr, 2, 53113 Bonn, Germany, Dec, 1993 (黄文奇)

NP wanquanxing lilun

NP 完全性理论 (theory of NP completeness) 研究 NP 完全问题的理论。一个问题被确认为具有 NP 完全性,或被称为是 NP 完全问题,意味着该问题是 NP 问题类中“最困难”的问题,是一个固有难解问题。对于这种问题,寻求一个现实可计算的(即多项式时间的)算法,是十分困难的,甚至可能是根本不存在的。历史上第一个 NP 完全问题是可满足性问题,它是由 S. A. Cook 于 1971 年提出的。下面是一个常被人们提起的 NP 完全问题。

旅行商问题 也称货郎担问题。假定一个销售员要到 n 个城市去推销产品。已知各城市间路程 d_{ij} ($1 \leq i, j \leq n$) 和一个界限 B , 问是否有一条旅行路线,恰好到每个城市一次,最后返回出发城市,且总路程不超过 B ?

旅行商问题还有另一种提法: 试求出总路程最短的旅行路线。前者称为判定形式问题,后者称为最优形式问题。可以证明两种形式问题在下述意义下是等价的: 如果对于最优形式问题有多项式时间算法,那么也容易找到对于判定形式问题的多项式时间算法,反之亦然。同时还容易看出,问题可以看成由无数个实例组成,一组 n, B 和 d_{ij} , 就代表了问题的一个实例。算法必须对问题的一切实例都能给出解答。

对于旅行商问题,从一个城市出发,有 $n-1$ 个城市可作为第二站,选定一个城市后,又有 $n-2$ 个城市可作为第三站,……。这样,总共将有 $(n-1)!$ 条不同的旅行路线。检查每一条旅行路线的总路程,就可以得到问题的解答。但这是一个指数时间复杂度的算法。指数时间算法被公认为不是现实可计算的算法。实际上,对一个底数为 2 的指数时间算法,当 n 为 50 时,即使采用每秒百万次运算的计算机,要计算出解答来,也需要 35.7 年;而当 n 为 60 时,则需要 366 个世纪,这是人们绝对无法容忍的。

一般 NP 完全问题,情况都与旅行商问题相似,都有一个比较明显的指数时间算法。多项式时间算法被公认为现实可计算的算法,那么 NP 完全问题是否有多项式时间算法? 这是 NP 完全性理论中的核心问题。

P 一类问题的集合。对类中任一问题,都存

在一个确定型图灵机 M 和一个多项式 p , 对于该问题的任何 (编码) 长度为 n 的实例, M 都能在 $p(n)$ 步内, 给出对这个实例的回答 (参见 **P 类问题**)。

NP 一类问题的集合。 对类中任一问题, 都存在一个非确定型图灵机 M 和一个多项式 p , 对于该问题的任何长度为 n 的实例, M 都能在 $p(n)$ 步内, 给出对这个实例的回答 (参见 **NP 类问题**)。

多项式时间多归约 假定给了两个问题 q 和 q_0 , 如果存在一个确定型图灵机 M 和一个多项式 p , 对于问题 q 中任意一个长度为 n 的实例 x , M 都能在 $p(n)$ 步内, 计算出问题 q_0 的一个实例 y , 使得 x 是 q 中有肯定回答的实例, 当且仅当 y 是 q_0 中有肯定回答的实例, 就称问题 q 多项式时间多归约到问题 q_0 。

NP 完全问题 如果 q 是 NP 类问题, 且 NP 中任意一个问题, 都可以多项式时间多归约到 q , 就称 q 为 NP 完全问题, 或称 q 具有 NP 完全性。在现实生活中已发现大量 NP 完全问题, 它们分布在计算机科学、数学、逻辑学和运筹学等诸多学科领域中。其中有代表性的, 除旅行商问题和可满足性问题外, 比较典型的还有划分问题, 带优先次序的处理机调度问题, 顶点覆盖问题和三维匹配问题等。NP 完全问题总数已达数千个 (参见 **NP 完全问题**)。

NP 完全性与 NP = ? P 问题 由于确定型图灵机是非确定型图灵机的特殊情况, 因而 P 类中任何一个问题都是 NP 类中的问题。有趣而重要的问题是其反问题: NP 类中任何一个问题是否都是 P 类中的问题? 如果是, 则 $NP = P$, 否则 $NP \neq P$ 。这就是著名的 NP 是否等于 P 的问题。它是计算机科学中一个非常重要而又历经 20 多年始终未获解决的问题。它的解决将导致一系列理论问题的解决, 还将关系到相当一批实际问题是否是“现实可计算的”这样一个大问题。

NP 完全性研究与 $NP = ? P$ 问题有密切关系。已经证明, 只要任何一个 NP 完全问题属于 P 类, 那么一切 NP 类问题都将属于 P 类, 于是 $NP = P$ 。实际上, 假定 q_0 是 NP 完全问题且 q_0 属于 P 类。现在考虑任意一个 NP 类问题 q 。由于 q_0 的 NP 完全性, 对于 q 的任给实例 x , 在多项式时间内可计算出 q_0 的一个实例 y , 而 x 是否有肯定回答取决于 y 是否有肯定回答。由于 q_0 属于 P 类, 于是可在多项式时间内判定 y 是否有肯定回答, 因而可以推出, 在某个多项式时间内, 可给出 x 是否有肯定回答的结论。如果要想证 $NP = P$, 只需找出一个属于 NP 类但不属于 P

类的问题。它理应是 NP 类中最难的, 即复杂度最高的问题。由于 NP 中一切问题都可以多项式时间多归约到任一个 NP 完全问题, 可以认为 NP 完全问题体现了 NP 类中各种问题计算中的困难性。因此, 如果 NP 类中真不属于 P 类的问题, NP 完全问题应该是首选对象。可以说, 如果证明了任何一个 NP 完全问题不属于 P 类, 则有 $NP \neq P$ 。遗憾的是, 迄今, 还未能找到任何一个 NP 完全问题具有多项式时间算法, 同时也未能证明任何一个 NP 完全问题不可能有多项式时间算法。即是说, 利用 NP 完全问题, 既没有证明 $NP = P$, 也没有证明 $NP \neq P$ 。

NP 完全问题的搜寻和证实 NP 完全性的相当一部分研究工作是去搜寻现实世界中可能有的 NP 完全问题, 并证明它们的 NP 完全性。当一个问题长久找不到多项式时间算法时, 就要认真考虑它是否是 NP 完全问题。很多 NP 完全问题正是这样得到的。证明一个问题是 NP 完全问题的主要途径是证明某个已知的 NP 完全问题可以多项式时间多归约到这个问题, 且该问题属于 NP 类。具体证明方法是多种多样的, 常用有限制法, 局部替换法以及分量设计法等。

NP 完全问题的求解 对于一个 NP 完全问题, 找到多项式时间算法, 就相当于证明了 $NP = P$, 而 $NP = ? P$ 问题历时 20 多年经许多人研究, 至今仍未解决, 这表明对 NP 完全问题寻找多项式时间 (即现实可计算的) 算法相当困难。另一方面, 如果 $NP \neq P$, 则表明 NP 完全问题根本不可能有多项式时间算法。然而, 面对现实生活中大量的 NP 完全问题, 人们必须找到有效的解决方法, 找到可以应用的算法。许多 NP 完全问题的明显算法是指数时间的穷举搜索法。

解决 NP 完全问题的一类方法就是采用各种方法, 尽可能减少这种搜索量。例如采用“分枝限界法”或“隐枚举法”, 它不能从根本上降低算法的时间复杂度, 但可以节约大量时间。此外, 还有动态规划法, 割平面法和拉格朗日法等。

第二类方法是处理最优化形式的 NP 完全问题。对于这类问题, 比较现实的态度是降低最优化要求, 求近似最优解, 以期得到一个多项式时间算法, 一个在容许时间内能得到容许精度的近似最优解的算法。这类方法常常是针对某类问题, 甚至是某个具体问题的。它们是一种探索式算法, 融会了设计人员的经验和智慧。在这些方法的研究中, 对

所得解和最优解近似程度的精确分析,以及如果 $NP \neq P$ 时各种 NP 完全问题的多项式时间算法在解的近似度方面的局限性的探讨,都是重要的研究课题。

与 NP 完全性研究有关的理论课题,除 $NP = ? P$ 问题外,还有 NP 结构的研究,多项式谱系的研究,多项式空间完全性的研究,以及对数空间复杂性研究等。

参考文献

1. Garey M R, Johnson D S 著. 计算机和难解性: NP 完全性理论导引. 张立昂, 沈泓, 毕源章译. 北京: 科学出版社. 1987
2. Hopcroft J E, Ullman J D 著. 自动机理论、语言和计算导引. 徐美瑞译. 北京: 科学出版社, 1986
(徐美瑞)

OBJ yuyan

OBJ 语言 (OBJ language) 用于书写和测试代数规约的可执行的形式语言。它采用抽象数据类型的方法刻画其语义,用重写规则系统实现。OBJ 由美国加州大学洛杉矶分校 J. A. Goguen 等人于 1977 年提出。

用 OBJ 书写的代数规约由若干对象组成,对象以 OBJ 开始,以 JBO 结束。每个对象由名、类别、操作、变量、等式五部分组成。名是标识该对象的惟一标识符。类别部分给出该对象引入及使用的类别,一个对象可以没有引入的类别。操作部分给出该对象定义的操作符的型构,即其参量类别和结果类别,等式部分刻画了这些操作符满足的性质。变量部分定义了等式中使用的变量的类别。

OBJ 除提供正常操作符外,还提供了异常操作符和恢复操作符,这三类操作符分别在 OK - OPS, ERR - OPS, FIX - OPS 中描述,共同构成操作符部分。异常操作符用于检测出错情况和形成出错信息,恢复操作符用于将异常出错值转换为正常值。

等式部分由 OK - EQNS, ERR - EQNS 和 EQNS 三类等式组成。OK - EQNS 刻画结果为正常值的等式,ERR - EQNS 刻画产生异常值(异常操作符)条件的等式,EQNS 中等式的结果可为正常值也可为异常值,用于刻画恢复操作符的性质。OBJ 解释系统中把等式看成自左到右的重写规则。

OBJ 允许用户定义操作符的语法,进行类别强制转换,允许操作符的重载。在对象的描述中,可以定义隐蔽操作符,它只在对象内可见,而对外不可见。此外,还可指定操作符满足结合律、交换律。对

出错情况的处理是 OBJ 的独到之处,通过提供异常操作符,可以比较精确地刻画出错条件及对错误进行定位。

在 OBJ 的基础上, J. A. Goguen 等人于 1987 年提出了 OBJ2。在 OBJ2 中,对一个已有的抽象数据类型,可定义新的操作和公理,对其模型进行扩充。此外,OBJ2 还提供了模块化机制。它们已用于书写机场调度的数据库系统等小规模实例。

参考文献

- Goguen J A, Tardo J J. An Introduction to OBJ: A Language for Writing and Testing Formal Algebraic Program Specifications. In: Gehani N, McGettrick A. Software Specification Techniques. Addison - Wesley Publishing Company, 1986
(费宗铭)

Occam yuyan

Occam 语言 (Occam language) 一种语句级并行的并行程序设计语言,用于描述多处理器互连网络上的并行算法及其实现。Occam 的主要概念是并行和通信,其主要思想源于 D. May 的实验性程序设计语言(EPL)和 C. A. R. Hoare 的通信顺序进程(CSP)。

Occam 没有类型概念,因而也没有变量说明。它的每个语句表示一个进程。简单语句描述原始进程;再通过进程构造组成复合进程,因此,程序也是一个进程。并行进程构造使多个进程可并行或并发执行。并行进程间的通信通过通道(channel)进行。输出进程

channel! expression

将表达式 expression 的值送到通道 channel 上;而输入进程

channel? variable

从通道 channel 上接收一个值并存入变量 variable。输入输出活动是配对的,输出进程的活动将被推迟直到对应的输入进程开始活动,反之亦然。

除了与常规的语句构造类似的顺序(SEQ),条件(IF)和循环(WHILE)进程构造外,Occam 还有并行(PAR)和选择(ALT)进程构造。例如,进程

PAR

channel? expression

channel! variable

表示并行地向通道 channel 上输出表达式 expression 的值,并从该通道上取出存入变量 variable,它等价于赋值语句 variable: = expression,但可用于两个处

理机之间的通信。选择类似于条件,但可依赖于其他进程是否正在进行输出。选择中的每一项由一个阀门后接一个进程。阀门由一个可选的真值表达式后接一个输入、延迟进程或空进程(**SKIP**)构成。例如:

ALT

```
red. selected&red. channel? x
out. channel! x
green. selected&green. channel? x
out. channel! x
```

NOT (red. selected **OR** green. selected)
&SKIP

```
out. channel! default. value
```

表示当 red 被选中并且 red 通道上有输出,则接收并送到输出通道上;当 green 被选中并且 green 通道上有输出,则接收并送到输出通道上;否则,向输出通道上送默认值。

为了表达时间概念,Occam 引入了局部时钟。时钟进程 `TIME? variable` 表示取当前时间,延迟进程 `TIME? AFTER expression` 表示进程挂起,直到时间超过表达式 `expression` 的值为止。

Occam 原来是作为大规模并行的 transputer 系统的“汇编语言”而设计的。它以其清晰、简洁的结构和同步机制使它代表了一种并行程序设计的风格。目前常用的是它的修订版本 Occam2。

参考文献

Jones C. Programming in OCCAM. Printice - Hall International (UK) Ltd., 1987 (伊波)

OpenGL tuxing biao zhun

OpenGL 图形标准 (OpenGL Graphics Standard) 以美国 SGI 公司的 GL 三维图形库为基础的一个通用共享的开放式三维图形工业标准。作为一个优秀的三维图形标准,OpenGL 提供了丰富的绘图命令,利用这些命令能够开发出高性能、交互式的三维图形应用软件。

目前, Microsoft, SGI, IBM, DEC, SUN, HP 等大公司都采用了 OpenGL 作为三维图形标准,许多软件厂商也纷纷以 OpenGL 为基础开发自己的产品,其中比较著名的产品包括动画制作软件 Soft Image 和 3D Studio MAX、仿真软件 Open Inventor、虚拟现实软件 World Tool Kit、CAD/CAM 软件 ProEngineer、地理信息系统软件 ARC/INFO 等。OpenGL 的稳定性、可靠性、易扩展性等特点,赋予了其强大的生命

力和应用前景。

OpenGL 实际上是一个开放的三维图形软件包,它独立于窗口系统和操作系统,以它为基础开发的应用程序可以相当方便地在各种平台间移植;OpenGL 使用简便,效率高。它具有如下 7 种功能:

(1) 建模 OpenGL 除了提供基本的点、线、多边形的绘制函数外,还提供了绘制复杂三维形体(球体、锥体、多面体等)及复杂曲线和曲面(例如 Bézier, NURBS 等曲线或曲面)的功能。

(2) 变换 OpenGL 的变换功能包括基本变换和投影变换。基本变换有平移、旋转、变比、镜像四种变换,投影变换有平行投影(又称正射投影)和透视投影两种变换。

(3) 颜色模型设置 OpenGL 颜色模型有两种,即 RGBA(A 表示 alpha 颜色分量)模型和颜色索引模型。

(4) 光照和材质设置 OpenGL 有辐射光、环境光、漫反射光和镜面光。材质是用光反射率表示的。场景中物体最终反映到人眼的颜色是光的红绿蓝分量与材质红绿蓝分量的反射率相乘后形成的颜色。

(5) 纹理映射 利用 OpenGL 纹理映射功能可以相当逼真地表达物体表面细节。

(6) 位图显示和图像增强 图像功能除了基本的复制和像素读写外,还提供融合、反走样和雾的特殊效果处理。以上 3 条可使被仿真物更具真实感,增强图形显示的效果。

(7) 双缓存动画 双缓存即前台缓存和后台缓存,后台缓存用来计算场景、生成画面,前台缓存显示后台缓存已生成的画面。此外,利用 OpenGL 还能进行深度提示、运动模糊等特殊效果的处理,并实现消隐算法。

参考文献

1. 乔林等. OpenGL 程序设计. 北京: 清华大学出版社, 2000
2. 中国游戏开发者. <http://mays.soage.com/develop/opengl/200112/GLInduction.htm> (张明敏)

OSI guanli tixi jie gou

OSI 管理体系结构 (OSI management architecture) 基于开放系统互连 (OSI) 环境对资源进行管理的一种体系结构。它提供了在开放系统互连环境中控制、协调和监视各种资源的手段。这些资源可以是开放系统互连环境中的各种实体和实体之间的连接等。图 1 示出了 OSI 管理体系结构。

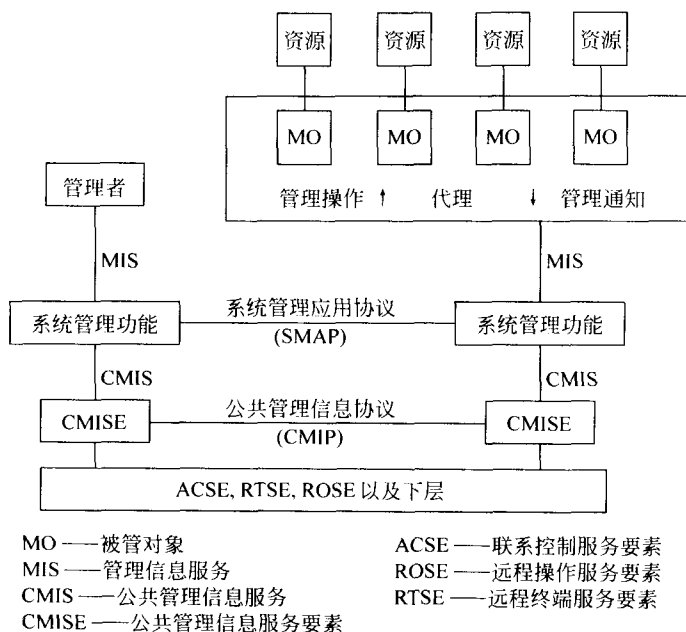


图1 OSI 管理体系结构

OSI 管理采用面向对象的方法来描述被管资源。被管对象(MO)是在开放系统互连环境中可以通过管理协议对其进行管理的资源的表现。被管对象的属性是资源的静态特性,而动作是资源的动态特性。被管对象的定义采用面向对象方法,对象类型可以有继承性,即除根对象外,其他对象均继承了其父对象类型的全部特征,包括属性、动作等。

OSI 管理向用户提供一系列管理信息服务(MIS)。系统管理应用协议(SMAP)是支持 MIS 的应用层协议。使用 MIS 的用户称为 MIS 用户。MIS 用户可以处于代理角色或管理者角色,或同时处于这两种角色。处于代理角色的 MIS 用户称为代理,处于管理者角色的 MIS 用户称为管理者。

代理可以对被管对象执行的管理操作有:创建对象、删除对象、动作、取属性值、置属性值、置默认属性值、增加成员、删除成员等。其中,前3个操作是针对被管对象本身的,其余则都是对属性的操作。

虽然被管对象的数量很大,但它们有很多共性。这些共性抽象出来,就组成了公共管理信息服务要素(CMISE)。CMISE 向系统管理应用协议(SMAP)提供公共管理信息服务(CMIS),采用的协议是公共管理信息协议(CMIP)。CMISE 提供的服务内容包

括:创建对象、删除对象、动作、置对象属性值、取对象属性值、对象事件报告等。

OSI 管理操作执行过程如下:首先,管理者向代理发出管理操作,要求代理完成一定的管理功能。然后,代理在收到管理操作后,对有关被管对象执行管理功能。最后代理向管理者发出管理通知。

管理信息库(MIB)是管理信息的集合,由于采用面向对象方法,MIB 实质上就是被管对象的有机组合,是由被管对象组成的管理信息树。

管理功能是由相对独立的一系列功能组成的。一个管理功能往往只起一种管理作用。已定义的管理功能有:对象管理功能、状态管理功能、关系管理功能、报警功能、事件报告功能、日志控制功能、安全报警功能、工作负荷监视功能、安全审计跟踪功能以及计费功能等。

参考文献

ITU - T Recommendation X. 700 (09/92) - Management framework for Open Systems Interconnection (OSI) for CCITT applications (钱松荣)

OSPF xieyi

OSPF 协议(Open Shortest Path First Protocol) 一种在 Internet(因特网)中采用的基于链路

状态路由算法的**内部路由协议**。它通过交换链路状态用图论的方法计算出最短的路径,并以此作为优先的**路由选择**。

Internet 的前身 ARPA 网最早使用的是距离向量协议,即 **RIP 协议**,1979 年开始采用链路状态协议。在 ARPA 网逐步发展成因特网后,1988 年因特网工程任务组(IETF)着手开发其后继协议。IETF 首先为该新协议提出了一系列的要求。第一条就是其算法必须发表在公开的文献上,这就是 OSPF 中“O”(Open,开放)的由来。该新协议在 1990 年成为因特网的标准,称为 OSPF。发表在因特网征求意见文献 RFC 1247 中,后来在 RFC 1583 和 RFC 2328 中又做了修订与补充。目前 OSPF 已得到广泛的应用,几乎所有的**路由器**产品都支持它。

OSPF 是一种链路状态路由协议,采用了链路状态路由算法(参见**路由选择**)的基本思想,即每个路由结点找到其邻点并测量和其连接的链路的状态信息,包括比特率、延迟时间和开销等;将链路状态组装成一个链路状态分组(LSP),发送给所有其他的路由结点;每个结点收到其他结点的 LSP 后就最终知道了完整的拓扑结构及每条链路的状态,然后可以按照一定的度量标准计算出最短的路径,建立路由表。

OSPF 中每个链路状态分组都以 IP 分组的形式发送。路由结点为了获取最新的链路状态信息,可以向邻点发送 OSPF 报文。报文共有 5 种,它们是:

- (1) Hello 用于发现谁是邻点;
 - (2) Link state update 为邻点提供发送者的开销;
 - (3) Link state ack 确认链路状态更新;
 - (4) Database description 通知发送者有哪些更新;
 - (5) Link state request 向邻点请求发送信息。
- 这些报文也是以 IP 分组的形式发送的。

在 OSPF 协议出现以前已有另一个著名的链路状态路由协议——IS - IS 协议。IS - IS 是最初为 DECnet 所设计的,后被国际标准化组织 ISO 所采纳。在因特网早期的主干 NSFNET 和数字蜂窝系统 CDPD 中都曾采用它作为**路由选择协议**。OSPF 比 IS - IS 晚几年设计,故吸收了后者的许多思想,如扩散链路状态更新信息的自稳定措施、支持分层次的路由和不仅计算最佳路径而且支持次优路径以利于分流和平衡负载等。

参考文献

Tanenbaum A S. Computer Networks. Third Edition. New Jersey: Prentice Hall, 1996 (高传善)

PARLOG yuyan

PARLOG 语言 (PARLOG language) 一种并行逻辑程序设计语言。PARLOG 的基本成分为单元子句(仅有一个正文字的子句)和条件子句(有一个正文字和若干负文字。正文字是原子公式,如: $p(x)$, 负文字是原子公式的否定,例如: $\sim p(x)$)。对于求单个解的谓词(也称单解关系)都有相应的 mode 说明,例如 mode $p(?, ^)$ 表示谓词 p 的第一个自变量为输入变量,执行前必须赋值,否则则延时执行。条件子句的子句中允许附加 Guard 条件,作为选择提交的依据之一。

PARLOG 的其他语言成分包括:能求解多解关系的集合谓词 Subset, Set; 顺序执行的或“;”操作和顺序执行的与“&”操作;三种合一原语。在语法上,除了 mode 说明和 Guard 条件之外,PARLOG 和 PROLOG 并无差别。

在行为上,PARLOG 完全不同于 PROLOG, 并行行为表现在能同时执行所有与目标。PARLOG 独有的特点是提供选择提交并行机制。具体地说,对每一个当前目标,如不满足 mode 说明,则延时执行,如满足 mode 说明,则并行搜索相关子句并同时检查其中的 Guard 条件,从中找出所有的候选子句并选出一个候选子句提交执行。这种独特的选择提交式并行执行机制保证了归结策略总是一直向前,没有回溯,直至成功或失败。

PARLOG 最早是由 K. L. Clark 提出的。此后,不少人又提出若干不同于 PARLOG 的并行语言,例如 1984 年 E. Y. Shapiro 提出 Concurrent PROLOG。

1990 年 K. L. Clark 进一步说明了 PARLOG 与其他并行逻辑程序语言的区别,提出另一种并行语言 LP。并行逻辑程序设计语言的最新进展是并发的约束 LP 语言。

参考文献

1. Clark K L. Parlog: Parallel Programming in Logic. Research Report DCC 84 /4 Imperial College, London, England

2. Clark K L. Parallel Programming. Computer Journal, 1990, 33(6) (胡运发)

PASCAL yuyan

PASCAL 语言 (PASCAL language) 在 **ALGOL 60** 语言的基础上发展起来的一种重要的程序设计语言。PASCAL 是 philips automatic sequence calculator 的缩略语,它是由 N. Wirth 设计的,1971 年正式发表。

1965 年 IFIP 工作组提出关于 ALGOL 60 后继语言的讨论。IFIP2.1 小组承接了提出其后继语言的任务,不久即分为两派。一派雄心勃勃想要在语言设计上另树丰碑,代表人物是 van Wijngaarden,由他设计了 ALGOL 68。而另一派认为 ALGOL 68 太复杂不易推广。N. Wirth 设计的 PASCAL 语言关心的是概念级、用户级和实现级的简明性。它已广泛地用于大学程序设计语言的教学和许多系统软件及某些应用软件的开发中。

其主要特点是:

(1) 丰富的数据结构和构造数据结构的方法。除了整型、实型、布尔型和数组外,还提供了字符、枚举、子域、记录、集合、文件、指针等类型。由这些数据结构可以方便地描述各种事务元。

(2) 简明灵活的控制结构。具体的结构语句有复合语句、如果语句、情况语句、While 语句、Repeat 语句、For 语句和处理记录变量的分量的缩写形式——With 语句。

(3) 它可称为第一个结构化程序设计语言。

(4) 编译运行效率高。

(5) 有利于书写程序设计语言的编译程序。

在 PASCAL 中没有引入模块的概念,不利于开发大型软件,这是它的不足之处。

BSI(英国标准化学会)1982 年正式出版英国标准 BS 6192(即 IS 07185),全名为“Specification for computer programming language Pascal”。我国 1987 年正式生效国家标准(编号为 GB 7591—87),全名为《程序设计语言 PASCAL》。1993 年公布修订标准 ISO/IEC 7185: 1990 文本。

参考文献

1. 郑国梁,钱士钧. PASCAL 语言——标准丛书. 北京: 中国铁道出版社,1989
2. Wirth N. The Programming Language PASCAL. New York: Springer-Verlag,1971 (郑国梁)

PDL yuyan

PDL 语言 (PDL language) 一种设计性语

言。它是由美国的 S. Caine 和 K. Gordon 在 1975 年提出的。PDL 是 program design language(设计性程序语言)的缩写,用于书写软件设计规约。它是软件设计中广泛使用的语言之一。

用 PDL 书写的文档是不可执行的,主要供开发人员使用。PDL 描述的总体结构和一般的程序很相似,包括数据说明部分和过程部分,也可以带有注释等成分。但它是一种非形式的语言,对于控制结构的描述是确定的,而控制结构内部的描述语法不确定,可以根据不同的应用领域和不同的设计层次灵活选用描述方式,也可以用自然语言。

PDL 语言书写的模块结构如下:

```
PROCEDURE<过程名>(<参数表>)
<数据说明部分>
<语句部分>
END<过程名>
```

数据说明部分形式为:

```
DECLARE<数据说明表>
```

数据说明表由一串说明项构成,每个说明项形如:

```
<数据项名> AS
<类型字或用户定义的类型名>
```

语句部分可以包括:赋值语句、if - then - else 语句、do - while 语句、for 语句、case 语句、调用语句、返回语句等。与一般程序模块不同,其语句中除描述控制结构的关键字外,书写格式没有严格定义。自然语言书写的注释可以插在任意位置。

下面是一个 PDL 描述的结构示例:

```
PROCEDURE
EXAMPLE
(arg 1, arg 2, ...)
DECLARE arg 1 AS
CHAR ARG
arg 2 AS
SCALAR ARG
...
<初始化操作>
DO WHILE<条件 1>
IF<条件 2>
THEN BEGIN
<具体操作>
END
ELSE<具体操作>
ENDIF
<具体操作>
```

ENDDO

RETURN

END EXAMPLE

PDL 描述接近自然语言(英语),易理解。它虽然不如图形化的设计描述直观,但和可执行的程序具有类似的结构,因此便于实现借助计算机自动转换为可执行的程序代码,已经研制出针对特定语言的自动工具。

参考文献

Caine S, Gordon K. PDL—A Tool for Software Design. Proc. National Computer Conference, AFIPS Press, 1975, 271 ~ 276 (陈道蓄)

PHIGS tuxing biao zhun

PHIGS 图形标准 (Programmer's Hierarchical Interactive Graphics System, PHIGS) ISO/IEC 制定的针对具有层次结构对象的交互式三维图形软件的国际标准 (ISO/IEC 9592—1, —2, —3)。与 GKS-3D 图形标准不同, PHIGS 能用于交互地建立、修改具有层次结构的形体模型并对其图形进行显示、管理。

从应用软件的角度看, PHIGS 表现为一组函数(或子程序), 可由 C 程序、FORTRAN 程序等来调用, PHIGS 定义的函数包括以下几类:

(1) 建模函数 使用 PHIGS 建模函数生成的形体模型以层次结构的形式来组织。每一结构元素可以是输出图元(参见图元生成)、属性说明、变换与裁剪、控制、编辑、通用和应用数据 7 种类型之一。由控制类结构元素将有关结构连成网络。PHIGS 还提供一组建模实用函数来形成建模变换所需的建模矩阵。

(2) 输出图元函数 PHIGS 输出图元有折线、多点标记、文字、加注文字、填充区、填充区集、单元阵列及广义图元 8 种。调用这些输出图元, 可以构造层次结构中的对象并由此生成其图形。

(3) 图元属性设定函数 用来设定输出图元的几何属性、非几何属性、观察属性及标识属性。

(4) 结构的编辑操作函数 提供对模型中各个结构元素个别进行存取和修改的手段。

(5) 结构的遍历与显示函数 把已经定义的结构或结构网络在工作站上进行显示, 并按结构优先级进行消隐(参见消隐技术)处理。

(6) 结构的查询与搜索函数 使应用程序能了

解已有结构各方面的情况。

(7) 结构的存档与检索函数 将已建结构或结构网络存入文件或从存档文件中取出。

(8) 观察操作函数 实现观察方向变换、观察投影变换、工作站变换以及裁剪操作。

(9) 交互功能函数 提供请求、采样、事件等 3 种输入模式, 定位、描画、数值输入、选择、拣取、字符串输入等 6 种逻辑输入功能。

PHIGS 集建模功能与图形显示于一体, 除几何数据外, 模型中还可包含大量应用数据, 有很强的交互能力。它适用于开发针对层次结构对象的二维或三维图形应用系统, 例如各类产品的计算机辅助设计系统。由于 PHIGS 实现时不分级别因而比较适用于大型应用系统。

PHIGS 标准的早期版本中有一个功能扩充部分 PHIGS PLUS (PHIGS +)。它在 PHIGS 的基础上增加了 11 种输出图元, 包括带数据的折线集、填充区、填充区集、三角形条、四边形网络、多面体、参数曲线与参数曲面、非均匀 B 样条曲线与曲面以及扩展的单元阵列等。这样, 除三维折线、区域、文字以外, 应用程序还可以使用新增加的输出图元来定义面片并构成三维形体。

PHIGS + 增加了绘制机制, 可以对描述物体表面的输出图元进行光照明处理。包括明暗处理、深度指示和颜色映射以生成逼真的图像。除了环境光以外, PHIGS + 支持方向光源、位置光源和聚光光源 3 种光源并区分几何法向量和反射法向量。PHIGS + 支持的明暗处理方法有不插值、颜色插值、法向量插值和点积插值(介于颜色插值和法向量插值之间) 4 种。PHIGS + 的深度指示按规范化设备坐标系中的 Z 坐标修改颜色, 以利于增强图形的真实感。PHIGS + 的颜色映射功能是为彩色图形输出设备而增加的。颜色映射方式有真彩色映射、伪彩色映射和三维伪彩色映射 3 种。PHIGS + 适用于三维形体的模型设计及其真实感图形的生成。

PHIGS 标准的 1997 年版本已经将 PHIGS 的基本功能与 PHIGS + 的扩充功能整合在一起, PHIGS + 的名称已不再使用。

参考文献

1. ISO/IEC 9592—1:1997, Information Processing Systems—Computer Graphics—Programmer's Hierarchical Interactive Graphics System (PHIGS)—Part 1: Functional Description (1997)

2. ANSI/ISO 9592—4, Information Processing Systems—Computer Graphics—Programmer's Hierarchical Interactive Graphics System (PHIGS)—Part 4: Plus Lumiere and Surfaces, PHIGS PLUS (1992)

3. 蔡士杰, 张福炎. 三维图形系统 PHIGS 的原理与技术. 南京: 南京大学出版社, 1992 (蔡士杰)

Phong ming'an chuli

Phong 明暗处理 (Phong shading) 对曲面物体或多边形网格的法向量采用插值方法进行光强计算的一种明暗处理的过程和方法。在应用问题中, 曲面物体常常近似地表示为一组多边形小平面片, 称为多边形网格。在用扫描线方法对这些小平面对面片的法向量进行线性插值, 从而在每一像素点上根据插值所得的法向量按光照模型计算光强 (参见光亮度计算)。算法的具体过程是: 首先计算多边形顶点处的法向量, 然后应用双线性插值求得每个像素处的法向量。如图 1 所示, 为了计算 P 点处法向量, 先分别用线性插值求得 E 处 (利用 A, B 点法向量) 和 F 处 (利用 C, D 点法向量) 的法向量, 然后利用 E, F 的法向量作线性插值求 P 处法向量。

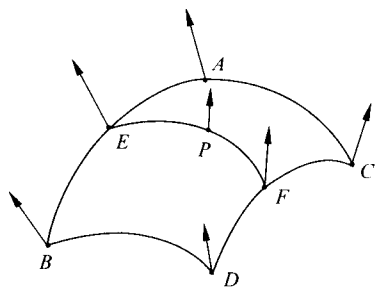


图 1 对法向量进行双线性插值

因为 Phong 明暗处理方法是向向量进行插值, 比 Gouraud 方法 (参见 **Gouraud 明暗处理**) 计算量大, 但在局部范围内对曲面表面光照的模拟更精确, 效果更好。

参考文献

1. 唐泽圣, 周嘉玉, 李新友. 计算机图形学基础. 北京: 清华大学出版社, 1995
2. 唐荣锡, 汪嘉业, 彭群生, 汪国昭等. 计算机图形学教程 (修订版). 北京: 科学出版社, 2001

(吴恩华)

PROLOG yuyan

PROLOG 语言 (PROLOG language) 一种顺序逻辑程序设计语言。PROLOG 是 programming in logic 的缩略语。PROLOG 程序有两类成分: 纯逻辑成分与非逻辑成分。纯逻辑成分由有限多个 **Horn 子句** 组成。Horn 子句是最多有一个正文字 (原子公式, 如 $P(X)$) 的一阶子句。单个正文字的子句称为事实; 有一个正文字和若干个负文字 (原子公式的否定, 如 $\sim P(X)$) 的子句称为规则; 仅有若干负文字的子句称为问题。纯逻辑成分的计算成分的计算和控制由 PROLOG 系统实现, 主要是合一、归结和回溯。

PROLOG 中非逻辑成分提供许多辅助功能控制特性, 例如: cut 操作用于控制, 允许用户干预搜索进程; 输入输出功能以及类似其他高级程序设计语言中的过程调用功能等。PROLOG 总是按书写的次序搜索规则操作符。“,” 表示顺序执行; “is” 表达式中所有变量必须赋值才能开始计值。

第一个 PROLOG 解释程序于 1972 年由法国马赛大学 A. Colmerauer 和 P. Roussel 小组实现。1977 年英国爱丁堡大学 D. H. D. Warren 实现了 PROLOG 编译程序, 执行效率较高, 达到实用水平。

参考文献

Clocksin W F, Mellish C S. Programming in Prolog. Springer-Verlag, 1981

(胡运发)

PSA chengxu

PSA 程序 (PSA program) 一种问题陈述分析程序。PSA 是 problem statement analyzer 的缩略语。PSA 可对用 PSL 书写的需求进行分析, 产生有用的文档。它是需求定义语言及其机器支撑方面的早期工作之一, 并已得到广泛使用。

PSA 是一种基于数据库的、用于支持 PSL 的交互式工具。它保存用 PSL 书写的软件需求, 并对之作语法检查和一致性检查, 从而形成一个由计算机管理的字典。PSA 可按照用户的需求对相应的字典进行查阅分析, 并可打出各类文档, 如格式化的问题描述、数据流图等。

参考文献

Teichroew D, Hershey E A. PSL/PSA: A Computer-Aided Technique for Structured Documentation and Analysis of Information Processing System. IEEE Transactions on Software Engineering, 1977, SE-3(1)

(吕建 陶先平)

PSL yuyan

PSL 语言 (PSL language) 一种问题陈述语言。PSL 可按一定的结构描述用户对软件系统的功能需求和性能需求。PSL 及其相应的支撑系统 PSA 最早出现在 1971 年,其第一个版本主要使用在美国密歇根大学的 IS DOS 项目之中,其主要目的是用于信息系统的需求定义与分析。它是需求定义语言及其机器支撑方面的早期工作之一,并已广泛使用。

PSL 从实体及其相互间关系的角度来刻画系统的输入输出、系统结构、数据结构、数据流程、系统规模、动态行为、系统性质、项目管理等各个方面。具体说来,在数据方面,PSL 提供了 ENTITY, CONSISTS OF, CONTAINED IN, DERIVED BY, UPDATED BY 等实体或关系来描述数据对象的名、数据结构和数据流程。在数据处理方面,PSL 提供了 GENERATES, RECEIVES, PROCEDURE, DERIVES, UPDATES 和 USES 等实体或关系来刻画处理的名、系统的输入输出、数据流程、处理方式和动态行为等。

严格说来,PSL 是一种半形式的语言,它在某些局部方面允许用户使用自然语言。例如,在描述处理时,可在 PROCEDURE 中用自然语言描述控制结构方面的信息。在描述数据时,可在 DESCRIPTION 中采用非形式的方式描述数据的各个方面。

参考文献

Teichroew D, Hershey E A. PSL /PSA: A Computer - Aided Technique for Structured Documentation and Analysis of Information Processing Systems. IEEE Transactions on Software Engineering, 1977, SE-3(1)

(吕建 陶先平)

P lei wenti

P 类问题 (class P of problems) 多项式时间内可解决的问题类,它是计算复杂性理论中十分重要的问题类。

在计算复杂性理论中,问题由无数个实例组成。考虑分割问题:给定 n 个自然数,是否可将它们分成两部分,使两部分自然数的和数彼此相等。不同的 n ,不同的自然数,就构成了不同的实例。

每个实例经过编码,可表成一个字符串。例如,将 n 个自然数表成二进制数,排成一行,中间用“#”号隔开,分割问题的每个实例,就表成字母表 $\Sigma = \{0, 1, \#\}$ 上的一个字符串。在这些字符串所代表的实例中,有些可以分成和数相等的两部分。

这些字符串就构成一个语言(参见形式语言)。

一个语言 L 称为属于语言复杂性类 P ,如果存在一个确定型图灵机 M 和一个多项式 p , M 在 $p(n)$ 时间内接受 L ,亦即对于每个长度为 n 的字符串 w ,只要 w 在 L 中, M 就能在 $p(n)$ 步内停机并接受 w 。

如果一个问题所对应的语言属于语言复杂性类 P ,就称它是 P 类问题。其直觉意义是:有一个算法和一个多项式 p ,若实例的编码长度为 n ,则该算法可在 $p(n)$ 步内给出问题的解答。正整数是否可被 4 整除的问题,是一个简单的 P 类问题。因为只要将该数表示成二进制数,图灵机扫视一遍这个输入,以确定最低两位数是否为 0,即可给出问题的答案。这个算法只需线性时间。现实中有大量 P 类问题。大多数数值计算问题都是 P 类问题。其他如排序问题,任务长度相等的多处理机调度问题,含删除、插入、替换操作的字符串到字符串的修正问题,箱容量固定的装箱问题和二维匹配问题等。

P 类问题公认为是计算机上现实可解决的问题。在表 1 中,假定每秒进行一百万次运算。容易看出,对于多项式时间复杂性函数,即使方次达到 5,当 n 增大时,计算时间平稳增长,且是人们可以接受的。而对于指数时间复杂性函数,即使底为 2,计算时间也随 n 急剧增长,常常达到人们无法容忍的地步。因此,人们公认,多项式时间可解问题是现实可解的问题。

表 1 多项式时间与指数时间复杂性函数的比较

复杂性函数	问题规模 n			
	10	30	50	60
n	0.01 ms	0.03 ms	0.05 ms	0.06 ms
n^3	1 ms	27 ms	125 ms	216 ms
n^5	100 ms	24.3 s	5.2 min	13.0 min
2^n	1 ms	17.9 min	35.7 年	366 世纪

参考文献

Garey M R, Johnson D S. Computers and Intractability. San Francisco:Freeman, 1979 (徐美瑞)

RIP xieyi

RIP 协议 (Routing Information Protocol)

一种最早在 ARPA 网中使用的采用距离向量路由算法的内部路由协议(参见路由选择)。RIP 中通常使用跳步数作为距离的度量标准。RIP 中允许出现的

最大跳步数是 16, 即从源到目标最多经过 16 个路由器结点。RIP 来源于 Bellman - Ford 路由选择算法, 由于通过 Berkeley UNIX 免费提供而流行起来。它最初用于 ARPA 网, 并收集在因特网征求意见稿 RFC 1058 中。早期版本的 DECnet 和 Novell 网的 IPX 也采用了它。

RIP 采用的距离向量路由算法与另一种常用的内部路由协议 OSPF 协议使用的链路状态路由算法相比较, 虽然在某些特定条件下存在收敛慢的问题 (即要花费很长的时间才能反映出网络中某些状态变化), 但在大多数情况下它都能稳定地工作。为此人们还提出了一些局部解决慢收敛问题的办法。更重要的是, RIP 协议相对简单, 因此仍然得到许多路由器产品的支持。

参考文献

Perlman R. Interconnections. Second Edition. MA: Addison Wesley, 2000 (高传善)

SETL yuyan

SETL 语言 (SETL language) 基于集合论的甚高级语言。它提供了描述有限集和元组及其有关操作和控制结构的手段, 以提高开发功效, 增加程序的易读性。该语言由美国纽约大学 J. T. Schwartz 等人于 1975 年提出。

SETL 在支持类 PASCAL 程序设计语言的大部分标准数据结构的基础上, 能够支持集合和元组。SETL 的集合必须是有限集, 元组的元素个数也必须是有限的, 但它们都可以任意嵌套。集合可以以枚举的形式给出, 也可以以 $\{x \text{ in } S | C\}$ (即集合 S 中满足条件 C 的所有元素构成的集合) 的形式给出。集合上的操作包括并、交、差、对称差、增添或删除元素、元素的个数、任取一元素等。集合上的谓词包括相等、属于、子集等。元组是元素的有序序列, 其操作包括并置、增添或删除元素等, 元组的谓词有相等、属于等。映射看成是二元组的集合。映射的操作包括取定义域、取值域、复合等。

SETL 语言支持描述软件规约的机制之一是它允许程序中使用全称量词和存在量词, 但这些量词中的约束变元必须受限于某一有限集, 即全称和存在的含义是就该有限集而言的。

SETL 的控制结构包括通常的条件、分情况、循环、转移, 还有其他与集合和元组有关的结构。与集合和元组有关的控制结构的基本形式为 (for v in $S | C$) SS end, 其含义是循环变量 v 依次取集合或元

组 S 中满足条件 C 的值, 执行语句 SS。 S 为集合时 v 取值的次序任意, S 为元组时依次取其元素的值。

SETL 允许定义通常的过程, 并支持模块定义及分别编译。SETL 提供一种表示语言, 程序人员可用来自指导 SETL 编译程序如何实现特殊的集合。SETL 编译程序通过分析输入的程序, 确定其中集合、量词、循环结构的实现方法。SETL 通过提供集合、元组、映射等复合结构, 增强了表达能力, 给用户的描述带来方便, 但代价是其编译产生的目标程序的效率较差。

SETL 语言以可移植的方式实现, 已在 IBM 370, DEC 10 和 VAX, CDC 6600 等机器上运行。在纽约大学、加州大学伯克莱分校等大学的课程教学中, 学生们已使用 SETL 语言进行算法设计。

参考文献

Schwartz J T, et al. Programming with Sets: An Introduction to SETL. New York: Springer - Verlag, 1986 (费宗铭)

Smalltalk yuyan

Smalltalk 语言 (Smalltalk language) 一种面向对象语言。1972 年由美国 Xerox 公司研究中心 (PARC) 以 A. Kay 为首的一个软件概念小组, 在 Flex 系统的基础上研制成功。后经不断的构思、试验和改进, 陆续推出若干版本, 其中最具影响的是 1981 年推出的 Smalltalk - 80, 但直至 1984 年才作为产品公开。

Smalltalk 有 5 个核心概念, 即对象、类、实例、消息和方法。对象是面向对象系统的惟一元素, 它的外部特征包括内部使用的若干私有变量和一组方法。类描述了性质相似的一组对象。类的每个对象称为该类的一个实例。消息是发送者 (对象) 传递给接收者 (对象) 的请求, 要求接收者执行所指操作。方法描述了操作的实现细节。

继承性是 Smalltalk 的特色, 它指的是, 子类继承父类的一切属性和操作, 整个系统的数据是通过子类机制组织成树型结构的。这种机制为信息共享提供了有效的支持。

Smalltalk 的基本语法结构是表达式。表达式是一字符序列, 它描述的对象称为表达式的值。Smalltalk 共有 4 种表达式:

(1) 文字表达式 它描述的对象是一个确定的常量, 即总是代表同一个对象;

(2) 变量名表达式 它描述的对象是可供使用

的变量,变量之值指该变量当前所指的对象;

(3) 消息表达式 它描述传送给接收者的消息,其值由该消息所引用的方法来确定;

(4) 块表达式 它描述的对象表示一系统被延迟的活动,常用来实现各种控制结构。

在 Smalltalk 中,建立程序就是根据类创建对象,执行程序就是不断向对象发送消息的过程。

Smalltalk 的主要特点是:①信息表示与处理的高度一致性;②弱类型语言;③比较完善的抽象机制;④语言融合于环境之中。

Smalltalk 在继承和并发等方面较弱。然而,它既推动了混合型 OO 语言(如 C++)的开发,又促进了对纯 OO 语言(如 Eiffel)的深入研究。自 1986 年以来,还推出了很多 Smalltalk 增强版本。例如,运行在微机 DOS 下的 Smalltalk /v,增加并行性的 Concurrent Smalltalk。

参考文献

1. Goldberg A, Robson D. Smalltalk - 80: The Language and Its Implementation. Addison - Wesley, 1983
2. Goldberg A. Smalltalk - 80: The Interactive Programming Environment. Addison - Wesley, 1983
3. 徐家福等. 对象式程序设计语言. 南京: 南京大学出版社, 1992 (郭浩志)

SNOBOL yuyan

SNOBOL 语言 (SNOBOL language) 一种串处理语言。SNOBOL 是 string - oriented symbolic language 的缩略语。SNOBOL 设计于 1962 年—1963 年,主要目的在于满足串处理的一般需求,另一重要的考虑是基于对符号化的数学表达式的处理。

在 SNOBOL 中串表示为字符序列。串括以引号,但引号并非串的组成部分。例如: 'Ruanjian'。这样的串特定表示为“字面常量”,串也可以用名来表示,如:

```
FIRST = 'Cheng'
```

它将串 Cheng 赋值给名 FIRST。串的字符个数不限,存储管理自动进行,无需进行说明定义。用串的并置表示并置运算,这种串可以用字面常量形式给出或用作某一个名的值,例如:

```
FULLNAME = FIRST 'San'
```

它将串 Cheng San 赋值给名 FULLNAME。为清楚起

见,空格用 ' ' 表示,也作为一种字符。

一个 SNOBOL 程序由语句序列组成,它有三种语句形式:赋值,模式匹配和替换。形式如下:

```
标号 主项 = 目标          goto
```

```
标号 主项 模式          goto
```

```
标号 主项 模式 = 目标      goto
```

标号标记语句,主项表示语句操作的对象,goto 控制程序执行流,且是任选的。赋值语句是将目标值赋给赋值号左边的变量名;模式匹配语句针对模式是否在主串中出现搜索制定的字符串;替换语句修改模式匹配的部分。

模式匹配是 SNOBOL 语言的一种最重要的基本功能。它是一种搜索主项中是否出现模式所指定的子串的过程。模式匹配是通过模式匹配语句实现的。

例如: Temp = 'Programming'

那么模式匹配语句

```
LAB1 Temp 'amm'
```

从左边开始扫描主项 Temp,搜索是否出现由模式指定的字符串 amm。如果指定的字符串在 Temp 上被找到,则模式匹配成功,反之模式匹配失败。因为在 Temp 中可以找到字符串 amm,上面的模式匹配是成功的。

带替换串的模式匹配语句是模式匹配和赋值相结合,先匹配一个子串,如果模式匹配成功再进行替换。语句

```
FULLNAME 'Cheng' = 'Zhang'
```

将子串 Cheng 用 Zhang 替换,它将 FULLNAME 的值换为 Zhang San。间接引用表示一个串可以进行运算,然后其值作为名使用。具体形式是以单目运算符 \$ 符后一个串的值作为名。例如:

```
X = 'NUM'
```

```
N = '2'
```

```
HOLIDAYS\AY = XN
```

```
$ HOLIDAY = 'The Spring Festival'
```

首先将 NUM 2 赋值给 HOLIDAY,然后将值 The Spring Festival 赋值给 NUM 2。这种间接引用操作,类似于汇编语言中的间接地址,它提供了一种在执行期间构造数据名的方式。

表达式是运算符和操作数的组合。运算符采用基本的算术运算符和串运算符,操作数为整或实数、字符串和变量。

GOTO 域出现在语句的最末位置上,它用来控制程序执行的转移。GOTO 可以是无条件地转向一个具有标号的语句,或是根据模式匹配的成功与失败条件地转向。用模式匹配的条件特征可以编制循环。

在 1965 年,SNOBOL 3 替代 SNOBOL。SNOBOL 3 类似 SNOBOL,但具有一些附加的特征,包括一些嵌入式函数和程序员自定义的递归函数的机制,SNOBOL 3 同样在 1967 年由 SNOBOL 4 替代,SNOBOL 4 成为广泛使用和实用有效的串处理语言。

参考文献

Griswold R E, Poage J F, Polonsky I P. The SNOBOL 4 programming language. Prentice Hall Inc., 1971 (郑国梁)

SPEC jizhun chengxu

SPEC 基准程序 (SPEC benchmark) 由 SPEC 开发的一组用于计算机性能综合评测的程序。

SPEC 是 system performance evaluation cooperative 的缩略语,它是由一些发达国家(主要是美国)的几十家世界知名计算机大厂所支持的一个非盈利合作组织,旨在开发共同认可的标准基准程序。以对 VAX 11/780 机的测试结果作为基数,其他计算机的测试结果以相对于这个基数的比率来表示。

1989 年提出了评测包括整数和浮点数运算在内的 1 组共 10 套基准测试程序,其中用于测量机器整数运算性能的程序共 4 套,称作 SPECint 89,用于测试浮点数运算性能的程序共 6 套,称作 SPECfp 89。

SPEC 92 是在 SPEC 89 的 10 套基准程序的基础上再增加了测试整数运算性能的程序,连同以前的 4 套合称 SPECint 92;增加了测试浮点数运算性能的程序 8 套,连同以前的 6 套共 14 套,称作 SPECfp 92。目前各计算机厂商生产的机器都分别标出 SPECint 92 和 SPECfp 92 的比率。

较新的 SPEC 92 的 20 套基准程序是基于不同的应用写成的程序。这些基准程序主要测量的是 32 位的中央处理器、主存储器、编译器和操作系统的性能。

6 套整数运算基准程序代表了以下的 application 范围:线路理论、LISP 解释器、逻辑设计、文本压缩算法、试算表以及软件开发。

14 套浮点运算基准程序(其中 3 套是单精度的)代表了以下的 application 范围:线路设计、蒙特卡罗模

拟、量子化学、光学、机械人学、量子物理学、天地物理学、天气预测和其他科学及工程计算。

1989 年的 SPECmark 评分方法(由 SPECint 与 SPECfp 混合而成)已不适用于日益发展的测试组合。新的 SPEC 92 测试组合的优点在于:

(1) SPECint 92 和 SPECfp 92 把基准程序分成了整数和浮点数两组,各有不同的工作负荷特性。

(2) 14 套浮点运算基准程序明显多于整数的 6 套。这是因为浮点运算在现实中涉及的应用范围明显多于整数应用。需要用更多的基准程序来说明机器的性能。

(3) SPECint 92 和 SPECfp 92 提供分开的评分结果,其优点是可让不同市场范围的顾客更了解产品的性能,克服了 SPEC 89 整数和浮点数混合评分带来的缺点。

SPECint 92 和 SPECfp 92 两组是分别进行测试的。测试方法是先将基准程序先在 VAX 11/780 机上运行,记下运行时间,求出其几何平均值;然后把把这些程序在被测机器上运行,也记下时间和求出几何平均值,用 VAX 11/780 机的几何平均值去除被测机器的几何平均值,测得的结果即为被测机器的 SPECint 和 SPECfp 值。SPEC 分值越高,说明性能越好。

这些厂商在 1995 年又推出了 SPECint 95 与 SPECfp 95 作为最新的测试标准程序。由于 SPEC 组织的成员都是世界知名的计算机厂商,所以 SPEC 基准程序的测试结果获得普遍的认同。(詹文岛)

TCP/IP xiejiji

TCP/IP 协议集 (TCP/IP protocol suite)

美国 ARPANET 和 Internet 等网络使用的通信协议。由于其可靠性高、用户广泛以及具有十多年的实际运行经验,TCP/IP 协议成为了事实上的工业标准。

TCP/IP 协议于 1983 年开始在 ARPANET 上运行,并于当年开始被插入 UNIX BSD 操作系统的内核,成为 UNIX BSD 操作系统的一部分。TCP/IP 协议随着 UNIX 操作系统的普及而广泛流行。现在,世界上的大部分国家和地区(包括我国)都已通过 TCP/IP 协议和 Internet 相连接。

使用 TCP/IP 协议的网络提供的主要服务有电子邮件、文件传输、远程登录和网络文件系统等。

基于 TCP/IP 协议的网络体系结构是指 TCP/IP 协议的层次关系和每层的协议。TCP/IP 协议及层次关系如图 1 所示。

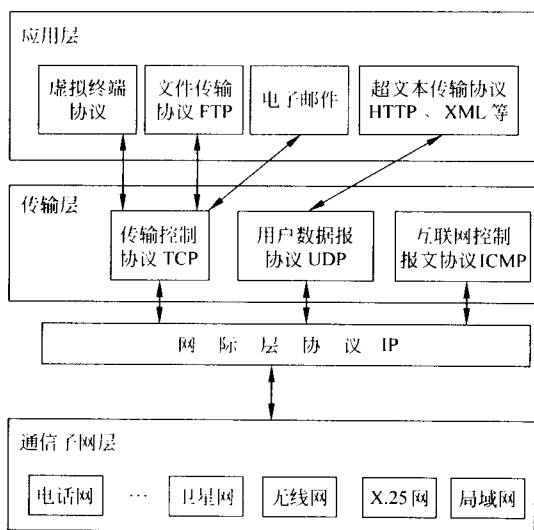


图1 TCP/IP协议及其层次关系

由图可知,TCP/IP协议实际上可分为四层,即通信子网层、网际层、传输层和应用层。

(1) 通信子网层 TCP/IP协议的通信子网层与OSI协议的物理层、数据链路层以及网络层的一部分相对应。该层中所使用的协议为各通信子网自己的固有协议,例如以太网的802.3协议和令牌以及分组交换网的X.25协议等。通信子网层的作用是传输经网际层处理过的消息。

(2) 网际层 网际层所使用的协议即IP协议,它把传输层送来的消息组装成IP数据包,并把IP数据包传递给通信子网层。IP协议通过提供统一的IP数据包格式,吸收了各通信子网的差异,从而为信息发送方和接收方提供了透明通道。

网际层的主要功能是:①Internet全网地址的识别与管理;②IP数据报路由功能;③发送时将IP数据包的长度分解为与通信子网所允许的数据包长度相一致,接收时对分解后的数据包进行重组。

(3) 传输层 传输层为应用程序提供端-端通信功能。传输层的主要协议有三个,即TCP、UDP和ICMP协议。

TCP协议以高可靠性地传输消息为目的。它负责把大量的用户数据按一定的长度组成多个数据包进行发送,并在接收到数据包之后按分解顺序重组或恢复用户数据。为了完成高可靠性数据传输任务,TCP协议具有数据包的顺序控制、错误检测、检验以及再发送控制等功能。

UDP协议提供无连接数据包传输服务。它把用户数据分解为多个数据包后发送给接收方。但是,UDP协议没有建立连接、数据包顺序控制、再发送和流量控制等功能,数据传输的可靠性由用户程序保证。UDP协议具有执行代码少,系统开销少和处理速度快等优点,特别适用于提供事务处理服务(参见用户数据报协议)。

ICMP协议是网际控制消息协议,主要用于端主机和网关以及网际网管理中心等的消息通信,以达到控制管理网络运行的目的。在传输的数据包有误或所传输的数据包丢失时,网关利用ICMP协议发送出错消息给发送有关数据包的端主机。另外,在数据包流量过大时,ICMP协议还有限制流量的功能。

(4) 应用层 应用层为用户提供用户所需要的各种服务。应用层的主要协议有:①远程登录协议rlogin;②文件传输协议FTP;③电子邮件协议SMTP;④支持万维网服务器的超文本传输协议HTTP等。

由于TCP/IP协议的提出先于OSI协议十多年,因此,TCP/IP协议的层次划分不同于OSI协议。OSI协议各层与TCP/IP协议各层的对应关系如图2所示。

OSI 协议		TCP/IP 协议	
7	应用层	4	应用层
6	表示层		
5	会话层		
4	运输层	3	传输层
3	网络层	2	网际层
2	数据链路层	1	通信子网层
1	物理层		

图2 TCP/IP协议和OSI协议的对应关系

TCP/IP协议经过了十多年的运行,已经变得相当成熟。但是,TCP/IP协议本身也受到了地址长度的限制,并且不能进行服务质量控制和传输多媒体信息。目前,许多关于TCP/IP协议的改进正在研究之中,例如RTP协议与IPv6协议等。

参考文献

张尧学等. 计算机网络与Internet教程. 北京:清华大学出版社,1999

(张尧学)

TPC jizhun chengxu

TPC 基准程序 (TPC benchmark program)

由 TPC 开发的评价计算机事务处理性能的测试程序。

TPC 是由硬件供应商、软件供应商以及用户组成的事务处理委员会 transaction processing council 的缩略语。TPC 旨在建立一套完整的基准程序以评测包括事务处理、数据库处理、企业计算与决策支持等广泛的商业计算。

TPC 于 1988 年 8 月成立,当时只有 8 个领先的系统和硬件公司参与,目前已有 40 多个成员,几乎包括了所有主要的商用计算机系统厂商和数据库厂商。TPC 于 1989 年 10 月首次宣布用于评测事务处理的 TPC - A 基准测试程序规范。目前已推出三代基准测试程序: TPC - A, TPC - B 和 TPC - C, 它们分别于 1989 年 10 月、1990 年 8 月和 1992 年 7 月发表。正在开发中的有 TPC - D 和 TPC - E。

TPC - A 基准程序规范用于评价在 OLTP (联机事务处理) 环境下的数据库和硬件的性能。不同系统之间用性能价格比进行比较。基准程序模拟了 1 个银行出纳员进行存款和出票时的网络,以对系统性能进行评测,用每秒完成的事务处理数 TPS 来表示。系统的价格包括硬件的购置和安装,5 年期内的软件租金和硬件维修等全部费用。性能价格比是以该价格除以系统的 TPS 值,并以每 TPS - A 需要多少千美元 (k\$ / TPS - A) 来表示。

TPC - B 测试的是不包括网络的纯事务处理量。系统仅由 1 个中央处理器、海量存储器和 1 个数据库组成,测试此系统每秒钟内所能完成的银行类型的事务处理的数量,而不考虑网络、终端等其他系统组成部分。由于 TPC - B 假设的环境意义不大,所以目前厂家的产品说明书不列出该项 TPS 值。

TPC - C 和 TPC - D 的对象是联机订货系统,如制造业、流通业以及商店的一般订货系统。它模拟了联机事务处理环境中的分销商的订单-输入应用环境。TPC - C 测试比 TPC - A 和 TPC - B 要复杂得多。它包括了混合的 5 种只读和更新密集的事务处理类型,系统全面地模拟了实际的应用。测试时,在保持对订单和其他 4 种事务处理类型作适当响应时间的同时,尽可能每分钟加入更多的订单。TPC - C 测试要求很苛刻,测试结果以事务处理每分 (TPM) 来衡量,而 TPC - A 和 TPC - B 是用事务处理每秒 (TPS) 来衡量的。TPC - C 是到目前为止,最好的基准测试程序,可以用来预测实际应用的性能。普遍

认为,TPC - C 是第三代也是最接近实际 OLTP 运行环境的标准测试程序。

TPC - D 的对象是决策支持系统。TPC - E 是模拟企业计算环境,其工作负荷的分配为: 2 / 3 是联机处理,1 / 3 是批处理。

TPC 基准测试程序在商业界建立了用于衡量商业性能和性能价格比的标准。

TPC 的成果是一系列已注册的基准测试规范。只有系统生产厂家根据指定的基准程序测试标准进行了测试,并向 TPC 提交了 1 份详细的完全公开的报告后,他们才可以在计算机市场上使用 TPC 商标。目前大多数计算机厂家的产品都在它的性能指标栏目标出 TPS 值。不过,由于处理性能随系统构成方式和所用数据库系统的不同而异,所以,不能简单地用给出的结果比较各种系统,具体问题还要具体分析。

(卢肖时 詹文岛)

Unicode bianma zifuji

Unicode 编码字符集 (Unicode Coded Character Set) 与国际标准 ISO/IEC 10646 完全兼容并同步发展的一种通用的字符编码标准。又称统一码或联合码。

自 1984 年国际标准化组织 (ISO) 启动通用多八位编码字符集 UCS 的项目后,初始的技术方案遭到了美国部分计算机公司的反对。1988 年初,美国 Xerox 公司的 Joe Becker 倡议将计算机字符集编码的基本单位由现行的 7 或 8 个二进位一举扩充为 16 位,充分利用 65 536 个编码位置以容纳全世界各种语言的字符和常用符号。新的字符集编码标准被命名为“Unicode”。IBM, DEC, Sun, Xerox, Apple, Microsoft, Novell 等公司共同出资成立 Unicode 协会 (The Unicode Consortium), 并由协会设立非赢利的 Unicode 公司,推动 Unicode 的开发工作。1991 年,1992 年出版了 Unicode 标准第一版 (The Unicode Standard, version 1.0) 的第一、第二册。

1991 年,在包括中国在内的各 ISO 成员体和信息领域的许多信息技术 (IT) 大企业的推动下,两大标准实现了相互对齐、合二为一。此后,这两个标准一直保持着协调关系、同步发展。例如,Unicode 3.0 版等同于 ISO/IEC 10646—1:2000, Unicode 3.1 版等同于 ISO/IEC 10646—2:2001。

总体上,ISO/IEC 10646—1:2000 与 Unicode 3.0, ISO/IEC 10646—2:2001 与 Unicode 3.1 的内容、编码、命名是完全相同的。但在一些细节上,两个标

准仍然有一些差别,比如:

(1) 10646 定义了四字节的编码空间结构(尽管绝大多数空间中还没有内容),支持四字节的编码表达方式 UCS-4;Unicode 目前只支持 UCS-2 和 UTF-16。

(2) 在汉字方面,10646 的文本采用多种字形,Unicode 只采用一种字形(非常接近中国的标准字形)。Unicode 提供汉字部首-笔画索引,10646 则不提供。

从组织上来讲,10646 是 ISO 制定的,ISO 以国家成员体为基础;Unicode 则是以公司为基础的集团制定的。

参考文献

1. <http://www.unihaan.com.cn/cjk/cjkhome.htm>
2. <http://www.unicode.org> (张福炎)

UNIX caozuo xitong

UNIX 操作系统 (UNIX operating system)

一种多用户交互式通用分时操作系统。由于它结构简练,功能强大,而且具有移植性、兼容性好以及伸缩性、互操作性强等特色,成为使用广泛,影响较大的主流操作系统之一,被认为是开放系统的代表。

发展简史

雏形阶段 UNIX 操作系统是 20 世纪 60 年代末由美国电报电话公司(AT&T)贝尔实验室的 Kenneth Thompson 和 Dennis Ritchie 于 1969 年实现的一种分时操作系统,最早的工作集中在文件管理和进程控制上,1970 年用交叉汇编方法,将该系统移植到 PDP-11 上,并提供给公司内部的专利部门用作文字处理。由于它吸取了以前的一个称作 Multics 系统的技术精华,又比 Multics 简单实用,开发者将它命名为 UNIX,这就是 UNIX 内核的雏形。

成型阶段 UNIX 设计者们继续进行功能扩展和版本更新,1972 年实现了极为重要的管道机制。1973 年 Ritchie 开发出 C 语言,它的出现是 UNIX 系统发展过程中的重要里程碑。用 C 语言改写后的第 3 版 UNIX 具有高度易读性、易移植性,为迅速推广和普及走出了决定性的一步。1974 年,“The UNIX Time-Sharing System”一文在美国杂志 CACM 上发表,引起广泛注意。最早外界可获得的 UNIX 是 1975 年的 UNIX 第 6 版。1978 年的 UNIX 第 7 版,可以看作当今 UNIX 的先驱,该版为今天 UNIX

的繁荣奠定了基础,UNIX 也步入了成型阶段。70 年代中后期 UNIX 源代码的免费扩散引起了大学和公司的兴趣,大众的参与为 UNIX 的改进、完善、传播和普及起到了重要的作用。最为著名的是美国加州大学 Berkeley 分校的 BSD 版本,其中加入了页式虚存管理、长文件名、快速文件系统、套接字、网络协议 TCP/IP 和 C-Shell 等大量先进技术,对 UNIX 的发展做出了很大贡献。

商业化阶段 1977 年 AT&T 公司开始为计算机软硬件厂商提供 UNIX 操作系统的初始设备制造商(OEM)许可证,商家开始了商业运营,许多商品化 UNIX 版本出现。比较著名的有: SUN 公司的 SUN OS 和 Solaris, Microsoft 公司的 XENIX, Interactive 公司的 UNIX 386/ix, DEC 公司的 ULTRIX, IBM 公司的 AIX, HP 公司的 HP/UX 和 SCO 公司的 UNIX 等。AT&T 公司本身则先后发展了 UNIX SYSTEM III、UNIX SYSTEM V, UNIX SVR4. 0、4. 1ES, UNIX SVR4. 2 等商品化版本。到 20 世纪 90 年代,不同的 UNIX 版本已超过 100 种。

标准化阶段 商业集团的参与促进了 UNIX 技术的迅速发展及普及,但也导致了版本繁多,互不兼容的局面。因此,从 20 世纪 80 年代开始,出现了对 UNIX 标准化的工作。UNIX 用户组织最早进行此项工作,后被美国 IEEE 接受和继承,并成立了标准化工作小组,着手制定基于 UNIX 的 POSIX(可移植操作系统界面)标准,到 90 年代初已有 20 多个 POSIX 标准正式颁布与制定。与此同时,UNIX 版权拥有者 AT&T 公司也在进行努力,1984 年颁布 UNIX SYSTEM V 的界面标准(SVID)。1988 年,AT&T 与 SUN 公司宣布联合开发 UNIX SYSTEM V 第 4 版计划,拟在兼容各主要 UNIX 版本基础上,使 UNIX SVR4 成为事实上的工业标准。此举得到了 Unisys、NCR、富士通等计算机厂商的支持,但却遭到 IBM、HP 和 DEC 等厂商的不满。他们联合成立了开放系统基金会(OSF)来抵制 SVR4 计划,而 AT&T 和 SUN 等公司成立了 UNIX 国际(UI)来推动 UNIX SVR4。UNIX 分裂为互为对抗的两大集团,这种分裂与竞争促进了 UNIX 技术的进步,但也延缓了 UNIX 市场的发展。

进入 20 世纪 90 年代后,由于多处理机和分布式网络处理技术的发展,UNIX 技术也在进一步的发展。UNIX 开始支持对称多处理机、图形用户界面、分布式处理,安全性也得到进一步加强,现已演变成为一种具有分布式处理能力的现代操作系统。

基本内容

UNIX 操作系统体系结构包含四个基本成分:内核、外壳(Shell)、文件系统和公用程序。

内核 是 UNIX 的基本核心。它负责调度和管理计算机系统的基本资源,包括进程、存储和各种设备的管理,以及实现进程间的同步和通信。进程管理包括进程的创建、调度、执行和撤销。存储管理包括内存、外存和虚存的管理。设备管理包括打印机、终端机、光盘机、磁带机和磁盘机等基本外设的管理。传统 UNIX 系统把基本文件系统包括在内核中;采用微内核技术的 UNIX 中,内核仅提供最基本的操作系统服务,其他如网络服务、文件服务等都移到用户空间中,以提高内核效率和便于内核移植。

文件系统 负责组织并管理数据资源。UNIX 文件系统采用树形层次结构,是一棵有根倒向树。最上端是根目录,第二层通常包括 etc、bin、lib 和 user 子目录。目录的层次可以不断扩充,而树枝是子目录,树叶为文件。可以通过路径名来访问目录和文件。早期 UNIX 仅支持一种字符串格式文件,目前 UNIX 已能支持包括“虚拟文件系统”在内的 10 多种文件系统,可以适应各种不同应用的需要。

外壳(Shell) 是一种命令式语言及其解释程序,命令语言是 UNIX 早期的用户界面。由于 UNIX 有管道(pipe)机制,而且提供了功能齐全的命令,加上 Shell 自身具有控制功能的语句成分,使得 Shell 命令语言功能相当强大,用户可在更高层次上进行程序设计,提高程序开发效率。Shell 的常用版本有 Bourne-Shell、C-Shell、Korn-Shell。WK-Shell 是具有图形开发能力的最新版本。

公用程序 或称工具软件,是 UNIX 系统提供给用户使用的常用标准软件,其内容相当丰富,包括编辑工具、管理工具、网络工具、开发工具、保密与安全工具等。目前所说的 UNIX 已经不再单纯指 UNIX 操作系统,因为它提供了丰富的工具软件,已经发展和演变成为一个功能强大的软件开发和运行环境。

主要特色

结构简练 以精巧的文件系统为代表。采用树形目录结构,文件的查找、增、删、改十分方便。文件系统可装卸,便于用户裁剪。外围设备都定义为特殊文件,统一以文件方式进行处理,简洁明了。

功能强大 除了常用功能外,UNIX 还首创了 pipe,它能将许多小功能片断连接组装、巧妙结合成复杂功能的软件工具。UNIX 也是最早具有创建异步进程能力的系统,早在 20 世纪 70 年代就实现了

多用户多任务功能。此外,UNIX 提供一系列的网络通信工具和协议,适用性好、使用广泛。

易移植性好 这一特色源于 C 语言和源代码开放政策。UNIX 系统 90% 以上代码用 C 语言编写,因此,易移植性好。到 1984 年,它已被移植到 70 多种计算机系列上。

可伸缩性和互操作性强 前者指在范围很宽的性能和配置的硬件上运行的能力;后者指在不同厂家的机器上运行和通信的能力。这两点是开放系统的基本特征,也是 UNIX 的重要特色,它从笔记本电脑到巨型机上都可以运行。迄今,已安装和运行在 100 多家计算机厂商的硬件平台上,这是其他操作系统无法比拟的优势。

完善的安全机制 由于 UNIX 基于多用户、多任务环境,设计了周到的安全机制,包括对用户的管理、对文件使用权限的管理、网络通信安全管理等。当今在世界各国许多关键性行业的信息化管理系统大都采用 UNIX 系统支撑。

发展趋势

UNIX 系统目前安装数量超过 500 万套,用户数达到 3000 万,已成为一种主流操作系统。从总体上看,UNIX 操作系统的主要发展趋势是统一化、标准化和不断创新。1993 年“公共开发软件环境(COSE)”组织成立,标志着主要 UNIX 厂商的联合和 UNIX 系统统一化的开始。同年,Novell 公司从 AT&T 公司购得的 UNIX 商标权无偿交给开放系统标准化组织 X/OPEN,这表明 UNIX 商标不再受某一厂商控制。在该组织的推动下,UNIX 的两个重要标准 Spec. 1170(标准应用程序界面)和 CDE(公共桌面环境)已于 1995 年正式颁布,为 UNIX 的统一化、标准化打下了重要基础。总之,由于 UNIX 的开放性,使它的发展充满活力和生机,与 UNIX 有关的新技术和新产品会不断涌现,UNIX 正是在这种既竞争,又协作的环境中不断发展和前进的。

参考文献

Muster J. UNIX 和 Linux 权威教程. 王玉馨,郑建超等译. 北京:清华大学出版社,2003

(贾耀良 费翔林)

USENET xinwen

USENET 新闻 (USENET news) 因特网上传播的最主要的电子新闻。电子新闻在因特网上随处可见,而 USENET 则是最主要的新闻传播工具。

电子新闻的传播是将信息从一台主机传送到另

一台主机,存储在电子邮件服务器中,当用户需要时,才传送给它。要阅读 USENET 新闻,主机用户只要运行新闻阅读程序,所订阅的所有新闻组的文章信息就会源源不断地显示出来,包括文章的作者、主题、第一页及后续信息。用户也可以发送信息给 USENET 新闻组。这些信息就是新闻。新闻组中的信息有些是全球性的,更多的是地方性和区域性的,涉及各种各样的话题。各个网络一般都作必要的取舍,只给网络用户提供相关的电子新闻。对用户而言,要获得最需要的新闻信息,可以对新闻进行 3 种不同层次的筛选:

- (1) 选择新闻组;
- (2) 选择文章;
- (3) 在显示文章第一页时,选择是否要继续阅读。

新闻组的增删决定于用户的选择,如果肯定的人多,一个新闻组就会创建或存在下去,否则,会被自动剔除。

(胡道元)

VAL yuyan

VAL 语言 (VAL language) 典型的数据流语言。VAL 是 Value-oriented Algorithmic Language 的缩略语。1979 年由美国麻省理工学院 (MIT) 的 J. B. Dennis 和 W. B. Ackerman 提出。它是一个小型研究性语言,其设计目标是为调整计算提供一个实验环境,以便人们了解数据流的计算过程。VAL 只提供了能表达算法固有并行性的最基本部分。它遵循以下两条设计原则:①提供隐式并行,任何指令级的并行均由计算机系统本身自动开发,而不需依赖语言的任何并行机制。VAL 的完全函数性支持了隐并行目标;②能有效转换为 DFG 和产生高效的程序。该语言简单明了,有助于程序人员了解计算系统的内部活动,进行错误处理、调试和效率分析。

VAL 的主要成分可分为说明部分和活动部分。详见表 1。

表 1 VAL 成分

说 明 部 分	基 本 类 型	· 整型 · 布尔型
		· 实型 · 字符型
	构 造 类 型	· 数组 · 记录
		· 择一
活 动 部 分	表 达 式	
	函 数	

VAL 具有很多强类型语言的特征,其值的类型和作用域都必须在程序中严格说明。每一类型,除了其定义值外,还包括相应的错误值。一旦出错,能迅速中止程序执行并作出处理。数组类型定义仅说明元素类型,不说明数组界限,界限直至构造数组值时才确定;择一类型的值在执行过程的不同时刻可有不同的类型。VAL 程序由一组模块组成,每一模块包含一个外部函数,可供其他模块调用。VAL 只有函数,没有过程。函数可嵌套。

为了适应数据流的需要,VAL 舍弃了若干强制式语言的特性(如变量、GOTO 语句),还提供了有助于描述数据流算法和各种表达式。

VAL 与 FP(函数式程序设计)在纯函数性、表达式、无副作用和程序代数规则等方面很相似,但二者也有明显的区别:VAL 有类型,FP 没有;FP 有函数型运算,VAL 没有。VAL 的主要缺点是:未提供通用的输入输出手段,缺乏递归性,未说明用什么方法实现结果收集。

参考文献

1. Ackerman W B, Dennis J B. VAL — A Value-oriented Algorithmic Language: Preliminary Reference Manual. Tech. Rep. TR — 218, Computation Structure Group, Laboratory for Computer Science. MIT, Cambridge, Mass, 1979
2. McGraw J R. The VAL Language: Description and Analysis. ACM TOPLAS, 1982, 4(1) (郭浩志)

VLIW chuliji

VLIW 处理机 (very long instruction word processor) 一种超长指令字处理机。其主要特点是采用显式并行指令计算方式。在 VLIW 处理机的一条指令(通常称为超长指令)中包含有多个操作字段,每个操作字段的功能相当于一般处理机中的一条指令,能够分别独立控制各自的功能部件同时工作。

VLIW 处理机的概念是由美国的 J. A. Fisher 于 1981 年首先提出, J. A. Fisher 等人领导的 Mutiflow 公司研制了世界上第一台 VLIW 处理机 TRACE28/300, 该处理机的一条超长指令中最多可以有 28 条可以同时执行的指令。

提出 VLIW 处理机的主要目的是要充分开发程序中的指令级并行性。目前,采用指令级并行方式的处理机主要有超标量处理机、超流水线处理机和 VLIW 处理机等三种。超标量处理机依靠设置多条

指令流水线,并通过同时发射多条指令来提高处理机的运算速度,而超流水线处理机则通过分时使用同一条指令流水线的不同部分的并发执行来提高处理机的运算速度。与超标量处理机和超流水线处理机相比,VLIW 处理机的主要特点如下:

(1) 显式并行指令计算(EPIC)方式 VLIW 处理机主要依靠并行编译器显式开发程序中的指令级并行性。在 VLIW 处理机上运行的程序可以看成是一个二维指令矩阵,指令矩阵中每一行上的所有指令组成一条超长指令,它们之间没有数据相关、控制相关和功能部件冲突,这些指令可以在 VLIW 处理机上同时执行。超标量处理机和超流水线处理机通常采用隐式并行指令方式,在这两种处理机上运行的程序是一维线性的指令序列,每条指令中一般只包含一个操作。在程序运行过程中,超标量处理机和流水线处理机只能在一个不大的指令窗口中通过硬件来寻找没有数据相关、控制相关及功能部件冲突的指令来并行执行。

(2) 指令级并行度高 超标量处理机和超流水线处理机的指令级并行度一般为 2 左右,通常不超过 4,而目前多数 VLIW 处理机的指令级并行度在 4 至 8 之间,有的已经达到几十。VLIW 处理机的指令级并行度高于超标量和超流水线处理机的主要原因是:由于在 VLIW 处理机中通过并行编译器来开发程序中的指令级并行性,可以在一个循环、一个函数、甚至整个程序中寻找指令级并行性,而且可以采用软件流水、循环展开等指令级并行度很高的方法充分开发程序中的多种并行性。

(3) 硬件结构规整、简单 VLIW 处理机主要由很规则的寄存器、存储器、运算部件和数据通路等组成,控制器简单,而且不需要复杂的指令并行调度窗口及多发射机制等。

(4) 编译器的实现难度大 VLIW 处理机的并行编译器主要依靠指令级并行算法、数据相关性分析算法、寄存器分配算法等来显式开发程序中的指令级并行性,从而提高处理机的运行速度。要研制指令级并行度高的编译器难度很大,执行代码的膨胀也很大。

许多计算机公司、著名大专院校都研制了自己的 VLIW 处理机,市场上也出现了多种 VLIW 处理机,其中,大部分是嵌入式、DSP、JAVA 虚拟机等专用处理机,其主要原因是,针对专用应用领域的 VLIW 编译器相对容易实现。已经推向市场的通用处理机有: Intel 和 HP 公司联合推出的安腾(Itani-

um)处理机,也称为 IA-64 处理机,IBM 公司推出的 DAISY 处理机,Transmeta 公司推出的 Crusoe 处理机等,Crusoe 处理机已经大量应用于笔记本电脑中,其功耗很低。其中,IA-64 处理机可以同时发射 6 条指令,因此也称为长指令字 L₆W 处理机,它有自己独立的系统软件和应用软件。

为了使 VLIW 处理机能够与其他通用处理机实现二进制兼容,IBM 公司推出了开放源代码 DAISY,它不仅可以实现 IBM 的 VLIW 处理器与 X86 处理机之间的二进制兼容,还可以实现 PowerPC、S/390、IBM 的 Java 虚拟机与 VLIW 处理器之间的二进制兼容。Transmeta 公司推出了代码映射软件,它可以使 Transmeta 公司的 VLIW 处理机 Crusoe 与 X86 处理机之间实现二进制兼容。(汤志忠)

VLSI 算法

VLSI 算法 (VLSI algorithm) 一类采用大量处理机处理、并考虑超大规模集成电路(VLSI)实现的并行算法。问题求解的并行计算过程与支持这一过程的计算结构相结合是此类算法的特点。VLSI 算法的两个重要性能指标是:时间复杂度(并行计算过程所耗费的时间)和面积复杂度(在超大规模集成电路上实现时相应计算结构所占的面积)。

VLSI 算法研究,主要是通过对问题计算规律的分析,挖掘出问题计算过程中可能存在的并行性,进而找出能充分利用这种并行性的并行算法以及支持这一并行计算过程的计算结构,最后分析算法的时间复杂度和面积复杂度。至今,已在代数运算(特别是矩阵运算)、图论问题、排序问题、数字信号和数字图像处理、逻辑推理和人工智能等方面,研究出了大量 VLSI 算法。例如,在超立方体结构上实现的 N 点 FFT(快速傅里叶变换)算法,其时间复杂度为 $O(\log n)$,而串行算法则为 $O(N \log N)$ 时间。又如,在三角形脉动阵列上实现的 n 阶线性方程组求解算法,其时间复杂度为 $O(n)$,而一般串行算法则为 $O(n^3)$ 时间。

在 VLSI 算法研究中提出了许多计算结构,例如线性阵列和环、二维网格和环网、多维网格、树、蝶型结构和超立方体结构等。图 1 示出了几种主要的计算结构。除了对这些典型结构进行的各种特性研究外,值得注意的还有两方面研究工作。一是关于实用结构的研究。例如,为解决二维网格上点间通信线路长(对 $N \times N$ 网格来说是 $O(N)$)和行列统计运算费时(一般要 $O(N)$ 时间)等问题,引进了树网结

构。为解决树结构中结点间通信不畅,特别是根结点附近信息堵塞严重问题,提出了X-树结构(树中同一级结点间加进水平通信线)和粗树结构(越接近根结点,通信线路和结点中处理通信的部件越多)。为解决超立方体结构中每个结点通信线较多的问题,提出了CCC网结构(在这种结构中,每个结点只有三条通信线)。另一类是各种结构之间关系的研究。其中最重要的一批结果是:线性结构(包括环)、二维网格结构(包括环网)、多维网格、树结构和树网结构等都可以嵌入超立方体结构,亦即前述结构都可以是某个超立方体结构的子图。此外,蝶型结构在某种意义上与超立方体结构等价。所有这些结果,对选择和设计大规模并行计算机的基本结构,都有重要意义。实际上,这类研究成果已经对国际上许多流行的大规模并行计算机基本结构的确定和不断演变,产生了实际的影响。

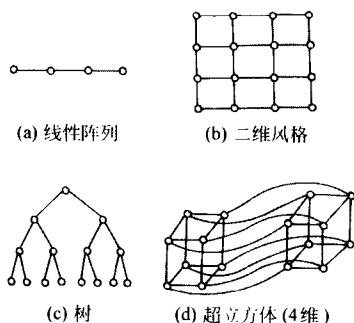


图1 几种主要计算结构

VLSI算法时间复杂性和空间(面积)复杂性的研究也是VLSI算法研究的重要内容。一些基本操作(例如加、减、乘、除和比较运算等)的时空复杂度是各种VLSI算法复杂度分析的基础,这方面已有一批结果。通常 k 位数加法的时间空间复杂度均为 $O(k)$,但也有时间为 $O(\log k)$,空间为 $O(k \log k)$ 的加法器。各种基本结构的VLSI布局,直接影响VLSI算法的面积复杂度。采用H-树布局方法,可使 N 个叶子的完全二叉树的面积由通常的 $O(N) \times O(\log N)$ 降到 $O(N)$ (图2)。关于下界的研究有助于对问题并行计算复杂性本质的了解,也有利于对问题的VLSI算法性能作出评估。由于VLSI算法的特殊性,必须同时考虑时间 T 和面积 A 两个因素。经常研究的有 A 下界, AT 下界和 AT^2 下界。已经证明, n 个 k 位数的排序问题,对于 $k \leq \log n, \log n <$

$k < 2 \log n$ 和 $k \geq 2 \log n$ 情况下的 AT^2 下界,分别是 $n2^k, n^2 \log^2(2^{k+1}/n)$ 和 $kn^2 \log n$ 。

在VLSI算法中,有一类算法受到人们的特殊关注,这就是脉动算法。脉动算法又称心动算法、搏动算法,是由H. T. Kung等人于70年代末首先提出来的。数据由主机不断地、有节奏地传送到处理机阵列中,数据在阵列中流动的同时,完成加工和处理任务,这恰似血液因心脏收缩而有节奏地流遍全身,在流动过程中完成其特定的生理功能。脉动算法有下述特点:①处理机具有简单的和一致的功能(阵列边缘上的处理机可略有不同);②处理机间有规则的和近邻的连接(如线性、四角或六角连接);③数据有规律和有节奏地流动;④数据在流动中被处理(流水线处理)。脉动算法的相应计算结构称作脉动阵列。

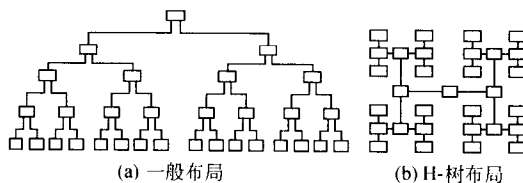


图2 结构的VLSI布局

图3是脉动算法的一个简单示例。这里是一个求矩阵矢量乘积的脉动算法,输入是一个 $n \times n$ 矩阵 A 和一个 n 分量矢量 b ,输出是一个 n 分量矢量 $c(c = Ab)$ 。脉动阵列中处理机的功能都相同,且很简单,只有乘积累加操作(图3(b))。处理机都是规则邻近连接,形成一个线性阵列。 A 矩阵数据按原有相对位置排列,呈倾斜状从上方流入阵列, b 矢量的分量则从左边依次流入阵列,这些数据在流动中不断参与运算(图3(a))。容易看到, n 拍以后,在 PE_1 中将得到 c 矢量的第一分量 c_1 ,而经过 $2n-1$ 拍,将在 PE_1, PE_2, \dots, PE_n 中分别得到 c_1, c_2, \dots, c_n 。

脉动算法已大量出现在以下各个领域:信号和图像处理、矩阵计算、多项式和整数运算以及包括排序、模式匹配、关系数据库等在内的非数值计算。

VLSI算法研究已经对算法专用芯片的研制、图像并行处理计算机系统的设计,以及对大规模并行计算的研究和大规模并行计算机系统的系统结构、系统软件和应用软件的研究产生重要影响。

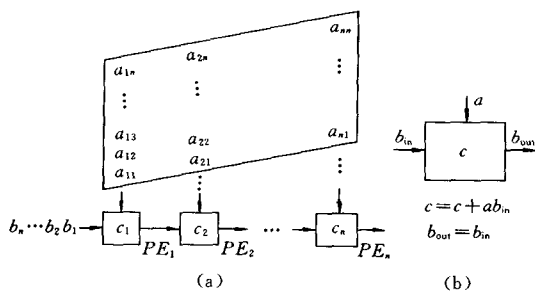


图3 脉动算法示例

参考文献

1. Ullman J D. Computational Aspects of VLSI. Computer Science Press, 1984
2. Kung S Y. VLSI Array Processors. New Jersey: Printice Hall, 1988 (徐美瑞)

Windows caozuo xitong

Windows 操作系统 (Windows operating system)

由美国 Microsoft 公司开发,支持多道程序运行,具有图形界面环境的操作系统。Windows 最初是作为对 DOS 操作系统的图形化扩充而推出的,已推出多个版本。它的多任务图形界面以及统一的应用程序接口使得在 Windows 环境下运行的应用程序的操作大为简化。Windows 获得了微型计算机操作系统的垄断地位。它在服务器软件市场也有相当建树。

发展简史

Microsoft 公司在 1983 年开始研发 Windows,其最初目标是在 DOS 操作系统的基础上提供一个多任务的图形化用户界面,并希望它能够成为基于 Intel x86 微处理芯片计算机上的标准操作系统。

继 1985 年和 1987 年分别推出 Windows 1.0 版和 Windows 2.0 版后,Microsoft 公司于 1990 年 5 月推出了 Windows 3.0。Windows 3.0 对 Windows 的内存管理、图形界面做了重大改进,使图形界面更加美观并支持虚拟内存,它以压倒性的商业成功确定了 Windows 系统在个人计算机领域的垄断地位。

随后,Microsoft 公司于 1995 年推出新一代操作系统 Windows 95,它对 Windows 3.x 版做了许多重大改进,包括:更加优秀的、面向对象的图形用户界面;全 32 位的高性能的抢先式多任务和多线程;内置对 Internet 的支持;更加高级的多媒体支持(声音、图形、影像等);即插即用;32 位线性寻址的内存管理和

良好的向下兼容性等。它可以独立运行而无需 DOS 支持,是操作系统发展史上一个里程碑式的作品。

在 Windows 95 取得巨大成功之后,Microsoft 公司继续对桌面版的 Windows 进行升级,Windows 98 在操作界面、联机帮助及辅助工具向导等方面都有了很大改进。它增加了用于自动检测硬盘、系统文件和配置信息的系统工具,内置了大量的硬件设备驱动程序,融合了当时最新的多媒体技术、网络技术。Windows Me 则是 Windows 98 的继续升级版,对用户提供更加强大的多媒体功能、高集成度的网络和更加友善的用户界面。

Windows 家族中另一重要的分支是 Microsoft Windows NT,它是由 Microsoft 发行的面向高端的操作系统。Windows NT 作为全新设计的操作系统,与原先支持个人应用的 Windows 有根本的区别。它采用客户-服务器与层次式结合的模式,体现了微内核结构操作系统 Mach 的思想,可以在从桌面系统到大型多处理器的网络服务器等一系列机器上运行。Windows NT 支持多进程并发,有较强的内置网络功能,系统安全性也达到较高水平。它所包含的 Win 32, Win 16, MS DOS, OS/2 和 POSIX 子系统提供了优越的应用程序兼容性,这一点是此前的任何操作系统无法相比的。

Windows 2000 是新一代个人计算机的商务操作系统。它建立在 NT 技术之上,具有高可靠性,高扩展性和业务优势,它通过简化系统管理降低了操作消耗,是一种适合从最小的移动设备到最大的电子商务服务器新硬件的操作系统。

Windows XP 是第一个把消费型操作系统和商业型操作系统融合为统一系统代码的 Windows,它结束了 Windows 两条腿走路的历史,是第一个既适合家庭用户,也适合商业用户使用的新型 Windows。

最新的 Windows 产品是 Windows 2003,它提供了联网、消息传递、集群、数据库到电子商务互联网(Web)站点以及文件和打印服务器等操作系统基础设施,具有高可靠性和高性能以及优异的商业价值。

到目前为止,Microsoft 推出了支持从个人数字助理、移动电话、接触式屏幕设备(如 Windows CE)到个人计算机、工作站、大型多处理器等的一系列 Windows 操作系统。

Windows 系统的构成

依据其提供的系统服务,Windows 系统主要由以下 3 个基本模块组成:

(1) 内核 内核实现对计算机资源的管理,并

提供系统服务和 Windows 的多任务管理,支持 Windows 应用程序所要求的低级服务,如动态内存分配、进程管理和文件管理等功能。

(2) 图形设备接口(GDI) 图形设备接口是一组图形设备驱动程序和库,是 Windows 图形功能的核心,它支持字体、绘图原语和用户显示及打印设备的管理。在此基础上,可实现 Windows 系统与设备无关的图形界面,并提供图形编程接口。

(3) 用户模块 用户模块实施对窗口的管理,且提供编程接口和外壳(Shell)功能。Windows 向用户提供两种类型的 Shell: 程序管理和文件管理,它们在形式上是一个窗口,用户对 Windows 的各种操作,都是在 Shell 窗口下进行的。

随着 Windows 的不断发展,Windows 系列的新产品在基本保持原结构的基础上,对上述 Windows 系统的基本功能模块作了相当程度的扩充和改进,以 Windows 2000 为例,从性能上的改进主要体现在如下:

(1) 可扩展性 Windows 2000 的可扩展性依赖于它的环境子系统。环境子系统向应用程序提供运行环境(操作系统功能调用接口),Windows 2000 有 3 个环境子系统: Win32、POSIX 和 OS/2 1.2,使得原先为这些系统开发的应用程序都可以在 Windows 2000 下运行。

(2) 易移植性 Windows 2000 通过硬件抽象层(HAL)将内核、设备驱动程序以及执行体同硬件分隔开来,使他们可以适应多种平台。从而提高了易移植性。

(3) 可靠性 Windows 2000 能主动地保护自身免受异常或外部有意或无意破坏的影响,并且对软件和硬件的错误做出可预测的响应。它的文件系统能自动从各种系统故障中恢复。

(4) 高性能 为获得高性能并进而得到系统的灵活性,Windows 2000 在系统设计中采用了一些好的算法和数据结构以及先进的通信机制,如本地过程调用(LPC)。

(5) 国际化 Windows 2000 基于 UNICODE,通过提供 NLS API,来支持不同地区的本地化使用。

主要特点

Windows 是系列产品,在它发展过程中的每一个新版本都有突出的新功能和新特点。它们已经对用户的工作方式和应用程序的开发产生了巨大影响,其影响力主要来源于贯穿整个产品系列的下列特点。

(1) 多任务的图形化用户界面 Windows 系统从一开始就摆脱了字符形式的操作界面,为每个运行的程序提供了一个独立的窗口,窗口的大小、位置、显示方式均可由用户控制,在窗口内分层次合理地组织了标题条、控制选单框以及各种按钮,除需要输入正文参数外,仅用鼠标器就可以方便地进行操作,执行各种功能。Windows 系统支持多任务,集中管理对应于每个任务的窗口,用鼠标器很容易在各窗口之间切换,实现多任务条件的切换。系统还支持动态数据交换,建立了任务间的数据联系。Windows 系统利用各种图示化手段,结合强大的联机帮助和提示机制,使得系统易学易用。

(2) 事件驱动的程序运行方式 Windows 支持基于消息循环的程序运行方式,使应用程序也采用类似操作系统的运行方式,而消息产生于用户环境引发的事件(如鼠标器或键盘动作)。与应用程序传统的序列驱动方式相比,事件驱动方式有较大的灵活性,对用户交互操作较多的应用程序有明显的优点。

(3) 标准的应用程序界面 Windows 系统为应用程序开发人员提供了功能强大的应用程序开发接口(API)。通过调用应用程序接口,开发者很容易创建 Windows 图形界面的各种元素,如窗口、选单、滚动条、对话框以及各种工具条等。其结果是应用程序在提供各自不同的功能时采用了风格一致的界面,也就是与 Windows 系统界面风格一致的界面。这不仅简化了应用程序开发,更重要的是大大简化了学习使用不同应用程序的过程。

Windows 系统还为应用程序开发提供了图形设备接口(GDI),实现了与设备无关的图形输出,使得应用程序能够以一致的方式调用同类设备。

(4) 不断增强的功能 每一种新版本的 Windows 都带来许多新功能,其目标始终是充分发挥不断增强的硬件能力以及尽可能更易于使用。突出的例子包括对内存的管理以及从 16 位升为 32 位,Windows 系统早已突破 DOS 对内存地址空间的限制。在 Windows 的不断发展过程中,系统逐渐集成了许多原先的工具软件,甚至某些应用软件的功能,在支持网络、多媒体、安全性等方面有了很大发展。同时保持了良好的兼容性,并不断提高用户界面的友善性。

发展趋势

从总体上看,Windows 系统今后发展主要趋势是功能更强大,安全性更高,使用更方便。

目前 Microsoft 公司正致力于 .Net 计划,试图通过使用分布式计算模型(如 DCOM)和基于开放标准(如 XML),允许用户应用程序通过 Internet 进行数据通信和资源共享。这一庞大的“无处不在的计算”计划,既是基于操作系统已有的成果,又对操作系统提出了更高的要求。Windows 的未来之路无论是继续其版本延续,还是以全新的面貌出现,都将服务于 .Net 计划。

参考文献

1. 尤晋元, 史美林等. Windows 操作系统原理. 北京: 机械工业出版社, 2001
2. Silberschatz A, Galvin P B and Gagne G. Operating System Concepts, Sixth Edition. John Wiley & Sons, Inc., 2002
3. Solomon D A and Russinovich M E. Inside Microsoft Windows 2000, Third Edition. Microsoft Press, 2000

(陈道蓄 茅兵)

X. 75 jianyi

X. 75 建议 (X. 75 Recommendation) 两个 X. 25 分组交换网互连时所遵循的协议标准(参见分组交换)。X. 75 建议和 X. 25 建议一样,都是由原国际电报电话咨询委员会(CCITT)制定的协议标准。CCITT 现已被改组更名为 ITU,它所制定的标准都称为建议。X. 75 建议和 X. 25 建议不同,X. 25 规定了作为数据终端设备(DTE)的计算机主机和作为分组交换网中结点机的数据电路设备(DCE)之间的接口。而 X. 75 则用来连接两个分组交换网,规定了两个被称为信令终端设备(STE)的装置之间的接口。这两个 STE 各在一个分组交换网中,通过 X. 75 连接后合在一起构成一个连接这两个分组交换网的完整的网络互连设备。图 1 给出了三个 X. 25 分组交换网通过 X. 75 建议互连的示意图。

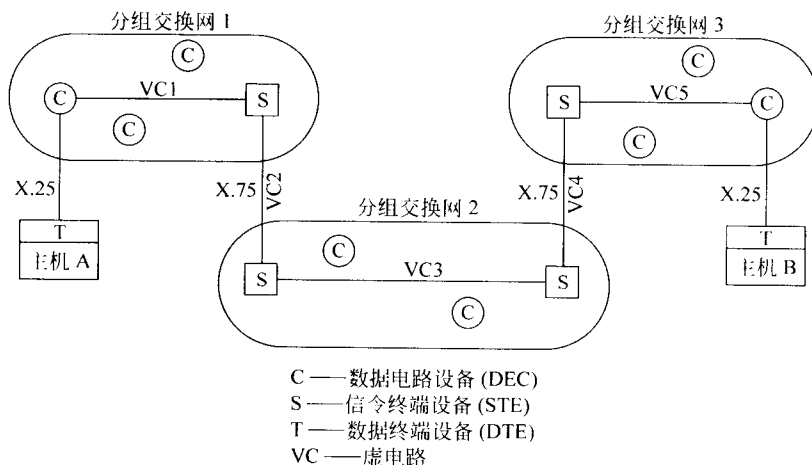


图 1 X. 25 分组交换网用 X. 75 互连

X. 75 和 X. 25 建议一样,都覆盖了开放系统互连基准(参考)模型的下 3 层。而且,X. 75 和 X. 25 一样也是以面向连接方式工作的,在这一点上它和 Internet 中广泛采用的网际协议(IP)是不同的。换句话说,通过 X. 75 建议传送数据前就和在 X. 25 分组交换网中一样必须先建立虚电路(VC)连接。图 1 中分组交换网 1 上的主机 A 和分组交换网 3 上的主机 B 之间是通过由 VC1、VC2、VC3、VC4 和 VC5 构成的虚电路通信的。其中,VC1、VC3 和 VC5 是网内虚电路;而 VC2 和 VC4 则是 STE 和 STE 间的网际虚电路,该两对 STE 间遵循的用于 OSI/RM 下 3

层的协议就是 X. 75 建议。从接入分组交换网的主机(数据终端设备(DTE))的角度看,是看不到 X. 75 的。DTE 和本网数据电路设备(DCE)的接口仍然遵循 X. 25 建议的规定,DTE 所看到的通过 X. 75 建议互连的分组交换网只是一个扩展的 X. 25 分组交换网。

参考文献

1. Tanenbaum A S. Computer Networks. Third Edition. New Jersey: Prentice Hall, 1996
2. 谢希仁. 计算机网络(第 2 版). 北京: 电子工业出版社, 1999

(高传善)

XCY yuyan

XCY 语言 (XCY language) 一种系统程序设计语言。由我国南京大学徐家福、中国科学院计算技术研究所仲萃豪、北京大学杨芙清等于 1978 年设计。XCY 语言规模适度, 简明易用, 主要用于书写系统程序。

XCY 语言是在开发 PASCAL、美国 DOD 红色语言及绿色语言等基础上, 兼顾书写系统程序的需要与简明易用两方面而设计的。XCY 语言提供的程序结构成分除通常子程序外, 还有模块和路径。

模块将逻辑上相关的对象(公用量、子程序和操作等)封装在一起, 通过模块说明引入。模块说明含一组说明、移出表和移入表。模块说明不可执行, 只有调用外部可见的子程序才能使用其中的数据和操作。XCY 模块可以嵌套, 可以分别编译。模块可以处于三种不同的工作方式: 管态、用户态、封锁中断。针对书写系统程序的要求, 模块有三种: 管程模块: 管理调度实资源; 类程模块: 控制作业路径专用的虚资源; 一般模块: 满足一般用户算法要求。

路径定义类似于子程序:

〈路径〉:: = path〈路径名〉〈形参部分〉〈路径体〉end〈路径名〉

〈路径体〉:: = 〈说明表〉〈语句表〉

但可以并发执行。

XCY 语言提供的数据类型包括以下 3 种:

- (1) 标准简单类型: 整型、布尔型、字符型;
- (2) 非标准简单类型: 字符串型、字位串型、子域型;
- (3) 构造类型: 记录、数组、联合。

XCY 语言中与机器有关的成分主要是用于指定数据对象的实际存储位置, 包括变量、模块位置信息以及记录中各个域的位置信息(包含这种信息的记录定义为 md 记录)。

DJS 200 系列计算机上的 RT 操作系统完全用 XCY 语言书写。XCY 语言还被用于书写不同的编译程序。XCY 本身的编译程序也是用 XCY 语言书写的。

参考文献

徐家福, 仲萃豪, 杨芙清. 系统程序设计语言 XCY 的设计. 北京: 计算机学报, 1981, 4(1): 1~12

(陈道蓄)

XYZ/E yuyan zu

XYZ/E 语言族 (XYZ/E language family)

一种系列化的时序逻辑语言族, 其中各子语言分别表示不同的程序设计方式或程序范型的语言。XYZ/E 由中国科学院计算技术研究所唐稚松于 20 世纪 70 年代设计, 它的最基本的特征是以一种统一的框架既能表示适应诺依曼体系的状态转换机制的命令式语言, 又能表示适应逻辑推理特征的直言式公式语言。

XYZ/E 有三种控制结构: 一种是直接表示状态转换的命令形式, 具有这种控制结构的子语言称为 XYZ/BE (即 Basic XYZ/E); 另一种则是结构化高级语言的语句形式, 具有这种控制结构的子语言称为 XYZ/SE (即 Structured rule form XYZ/E); 第三种控制结构则是产生式规则的形式, 具有这种控制结构的子语言称为 XYZ/PE (即 Production rule form XYZ/E)。

XYZ/E 中也包括了表示各种并发性或不确定性、不同通信方式、不同类型的可重用模块的机制。故在一统一的程序中可包含所有这些机制及相应的各种程序设计方式。它同时还能包含表示多种可视图形程序的语义, 而且这些图形与相应的 XYZ/E 程序可相互自动生成。由可重用模块(过程、进程、包块)与并发通信机制结合而成的程序, 由于其结合方式不同, 可以构成差异很大的总体结构, 其中有些情况是互不相容的。可以区分为三种类型: ①非分布式环境下基于对象的程序; ②非分布式环境下面向对象的程序; ③分布式程序。

XYZ 系统是将时序逻辑与软件工程有机结合、基于 XYZ/E 语言的计算机辅助软件工程(CASE)环境。故它构成一正交的二维体系。一维是基于 Manna-Pnueli 线性时间时序逻辑语言族 XYZ/E, 另一维是 CASE 工具集, 包括五组工具(交互式验证工具与自动生成工具、记录历史的逐步求精与原型速成工具、结构化设计的可视图形工具、语言转换工具、软件管理工具), 均以 XYZ/E 表示其语义界面, 它们既可独立使用又可根据其输入输出界面语义一致性相互连接组成更复杂的工具。

(柳军飞 赵琛)

Z yuyan

Z 语言 (Z language) 一种以状态机为模型的形式规约语言。最早由法国人 J. R. Abrial 提出, 由

C. A. R. Hoare(英国牛津大学)领导的程序设计研究小组发展而成。

Z的基本单位是模式。一个模式用一右边开口的矩形框起,中间一道横线将它分成说明部分和谓词部分。说明部分定义一些状态或模式变量;谓词部分是一般的谓词公式,它给出了变量间的限定关系。模式可定义系统的状态空间,初始状态和状态变换。如下面的模式定义了公寓出租系统的状态空间:

```
RentApart
apartments: P APARTMENT
customers: P PERSON
booked: APARTMENT  $\rightarrow$  TIME  $\rightarrow$  PERSON

dom booked  $\subseteq$  apartments
 $\forall (ap: APARTMENT | ap \in apartments.$ 
  ran booked  $\subseteq$  customers)
```

其中 RentApart 是模式名。apartments 是一个公司所有的公寓;customers 是该公司的客户集。booked 是一个部分函数,它指明某一公寓在特定时间里租给了某个客户。谓词部分指出,booked 的定义域(公寓)必须是该公司的财产,而租用人(booked 的值域)必须是该公司的客户。说明和谓词部分都可以有多行,行与行之间分别是“拼接(;)”关系和“与(\wedge)”关系。模式也可以线性地表示,如上面的模式可表示为:

```
RentApart  $\equiv$  [ apartments: P APARTMENT; customers: P PERSON;
  booked: APARTMENT  $\rightarrow$  TIME  $\rightarrow$  PERSON
   $\wedge$  (dom booked  $\subseteq$  apartments)  $\wedge$ 
   $\forall (ap \in APARTMENT | ap \in$ 
    apartments  $\wedge$  ran booked  $\subseteq$  customers)
  ]
```

由此可见,一个模式定义了一个(子)集合。作为集合,模式可以进行诸如并(\vee),交(\wedge),蕴含(\supseteq),幂集(P),大小($\#$)等集合运算,也可以移入已定义的模式。

模式既可以表示数据状态,也可以表示运算。运算通过状态变化来表示。当用来表示运算时,用 ΔS 表示模式 S 的状态改变,用 a' 表示变量 a 改变后的状态值。如:

```
Booking
 $\Delta$ RentApart
a?: APARTMENT
t?: TIME
p?: PERSON

a?  $\in$  apartments
p?  $\in$  customers
t?  $\notin$  dom booked a?
booked' = booked  $\cup \{a? \rightarrow t? \rightarrow p?\}$ 
apartments' = apartments
customers' = customers
```

表示,当一间公寓($a?$)某个时间($t?$)未租出,则将它租给客户($p?$)。这里,变量后面的“?”表示该变量是输入变量;若要表示输出,则在变量后面加“!”。

一个 Z 规约就是一组模式和用自然语言书写的对各模式的注释。模式之间通过模式运算和移入相关联。

Z 以其直观、简洁、接近功能分解模型吸引了众多的用户。利用谓词演算和模式演算的数学性质,可对 Z 规约进行形式分析。现已有各种 Z 规约辅助开发工具。针对面向对象的软件风范,Z 又有了 Object Z 和 Z++ 等扩充。

参考文献

1. Spivey J M. Introducing Z: A Specification Language and its Formal Semantics. New York: Cambridge University Press, 1988
2. Spivey J M. The Z notation: A reference Manual. New York: Prentice Hall. Inc., 1989 (伊波)

α - β jianzhi

α - β 剪枝 (alpha - beta pruning) 博弈树搜索中寻找一步好棋的有效过程。它的第一个版本由 A. Newell, J. Shaw 及 H. Simon 于 1958 年提出。 α - β 过程的基本思想是在树生长的同时进行端节点估值和倒推值计算,以修剪树中与一步好棋无关的子树,从而改善极小极大过程中搜索树生成和位置估值相分离导致的低效率。 α - β 过程如下(其中 f 是估值函数):

函数 Alphabeta(p : 位置; α : 整数; β : 整数)

Var m, i, t, d : 整数;

begin

确定位置 p 的后继 p_1, \dots, p_d ;

如果 d 为 0, 则 $\text{Alphabeta} : = f(p)$;

否则, begin

$m : = \alpha$;

for $i : = 1$ to d do

begin

$t : = -\text{Alphabeta}(p_i, -\beta, -m)$;

如果 $t > m$, 则 $m : = t$;

如果 $m \geq \beta$, 则转到 done

end

done : $\text{Alphabeta} : = m$

end

end

在搜索期间, α 与 β 值计算方法如下:

(1) 一个 max 节点的 α 值等于其后继节点当前最大的一个最终倒推值;

(2) 一个 min 节点的 β 值等于其后继节点当前最小的一个最终倒推值。

由于 max 节点的 α 值单调增加, min 节点的 β 值单调减少, 因此若任意 min 节点的 β 值不大于它的任意 max 祖先节点的 α 值, 则进行 α 剪枝, 即中止该 min 节点以下的搜索, 并设置该节点的最终倒推值为其 β 值; 若任意 max 节点的 α 值不小于它的 min 祖先节点的 β 值, 则进行 β 剪枝, 即中止该 max 节点以下的搜索, 并设置该节点的最终倒推值为其 α 值。当得到初始节点的全部后继节点的最终倒推值时, α - β 过程结束, 而最好的一步棋应走向具有最大倒推值的后继节点。

在 α - β 剪枝中, α 与 β 值是基于端节点的静态值计算, 需采用深度优先方法对搜索树的某一部分生长到最大深度, 一次搜索中的剪枝数取决于早期的 α , β 值与最终倒推值之间的近似程度。初始节点的最终倒推值将等于某一端节点的静态估值, 因此若在搜索中第一次就遇到该端节点, 则剪枝数最大, 需要生成和估值的端节点数最小, 即最佳 α - β 剪枝。

下面通过实例给出 α 与 β 剪枝情况, 图 1 表示 α - β 过程已获得节点 P_1 , 倒推值为 4, 并在 P_2 设置 α 等于 4, 当生成和估值 P_{21} (值为 2) 后, 我们知道节点 P_2 倒推值的上界不会比 2 大, 而节点 P 的倒推值不会比 4 小, 因此就可不必对 P_2 以下的节点继续进行搜索, 即发生了 α 剪枝。图 2 表示 α - β 过程已获得节点 P_{11} , 值为 4, 并在 P_{12} 设置 β 为 4, 当生成和估值 P_{121} (值为 8) 后, 我们知道节点 P_{12} 倒推值的下界不会小于 8, 而节点 P_1 的倒推值也不会大于 4, 因此

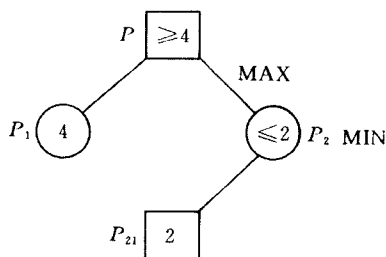


图 1 α 剪枝

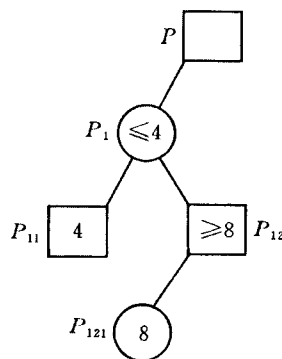


图 2 β 剪枝

可不必对 P_{12} 以下的节点继续进行搜索, 即发生了 β 剪枝。

设一棵树深度为 d , 且每个节点含 b 个后继节点, 则深度优先的极小极大过程将生成树中全部 (共 b^d 个) 端节点; 而最佳 α - β 过程仅生成 $2 \times b^{d/2} - 1$ (当 d 为偶数) 或 $b^{(d+1)/2} + b^{(d-1)/2} - 1$ (当 d 为奇数) 个端节点, 即对深度为 d 的最佳 α - β 过程搜索所生成的端节点数相当于深度为 $d/2$ 的极小极大过程搜索生成的端节点数。在最坏情况下, α - β 过程生成全部端节点。

参考文献

1. Kumar V, Nau D, Kanal L. A General Branch-and-Bound Formulation for AND/OR Graph and Game Tree Search. In: Kanal L, Kumar V. Search in Artificial Intelligence. New York: Springer-Verlag, 1988. 91~130

2. Nilsson N J 著. 人工智能原理. 石纯一等译. 北京: 科学出版社, 1984

(怀进鹏)

λ yansuan

λ 演算 (λ calculus) 一种研究函数的定义与

求值的一般方法。它是一种等价于图灵机理论的计算模型。λ 演算既是函数式程序设计语言的数学模型,又是程序设计语言的指称语义的理论基础。λ 演算与直觉主义逻辑相结合为交互式证明编辑系统等软件工具提供了逻辑框架。

第一个 λ 演算系统是由美国学者 A. Church 为解决数学基础问题提出的。A. Turing 和 J. C. Kleene 证明了图灵机递归函数与 λ 演算作为计算模型的等价性,导致了图灵-立奇论题。美国学者 J. B. Rosser, H. Curry 等都对 λ 演算的发展做过重要贡献。1969 年 D. Scott 发展了 λ 演算的模型论,并与 Strachey 合作给出了程序设计语言的指称语义。20 世纪 60 年代 J. McCarthy 设计并实现了第一个以 λ 演算为模型的程序语言 Lisp。1978 年 J. Backus 倡导使用基于 λ 演算与组合逻辑的函数式程序设计语言,推动了程序语言的研究和发展。80 年代初 P. Martin-Löf 将直觉主义逻辑思想引入计算机科学,使用带类型 λ 演算,给出了通过对程序规约进行证明,而直接构造程序的开发方法,进一步扩大了 λ 演算在程序理论中的应用,也推动了高阶带类型 λ 演算的研究。

λ 演算的主要理论包括它的语法范畴,归约理论,模型论以及带类型 λ 演算等几个部分。

λ 项是 λ 演算的基本对象。令 V 为变元组成的可数无穷集合,λ 项可以递归定义如下:

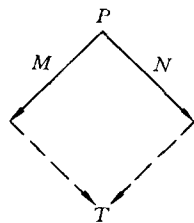
- (1) 若 $x \in V$, 则 x 是一个 λ 项。
- (2) 若 M, N 都是 λ 项, 则 MN 是一 λ 项。
- (3) 若 M 是 λ 项而 $x \in V$, 则 $\lambda x. M$ 是一 λ 项。

例如 $xx, \lambda x. x, \lambda x. xy, (\lambda x. x)(\lambda x. x)$ 等都是 λ 项。形如 $\lambda x. M$ 的 λ 项称为一个抽象, x 称为约束变元, M 是 x 的作用域。它大体相当于一个程序设计语言的过程说明,其中 M 相当于过程体, x 相当于形参, λ 相当于过程说明标记。 MN 称为一个作用,它相当于一个过程调用, N 相当于实参。不同的是在 λ 演算中 M 和 N 可以是任意 λ 项。

若变元 y 不在 M 中自由出现, 则用 y 替换抽象 $\lambda x. M$ 中 M 的自由变元 x , 记成 $\lambda y. [y/x]M$, 称为对此 λ 项的一个 α-转换, 如果 M 是由 N 经有限步 α-转换而得, 则称 M 与 N 是同余的。这里 $[y/x]M$ 表示用 y 替换在 M 中 x 的所有自由的出现。 $(\lambda x. M)N$ 称为 β-可约项, 而相应的 $[N/x]M$ 称为一个约减项。若一 λ 项不包含 β-可约项, 则称此项为 β-典范式, 将一个 λ 项中的 β-可约项用它相应的约减项替换称为对此 λ 项的一次 β-归约。若 λ 项 N

是由项 M 经有限次 β-归约及 α-转换而得, 则称 $M\beta$ -归约到 N , 记为 $M \rightarrow_{\beta}^* N$ 。如果存在 M_0, M_1, \dots, M_n , 使 $M_0 \equiv M, M_n \equiv N$, 且对任何 $i < n$ 有 $M_i \rightarrow_{\beta}^* M_{i+1}$ 或 $M_{i+1} \rightarrow_{\beta}^* M_i$, 则称 M 与 $N\beta$ -相等, 记为 $M = N$ 。可以证明: 对任意 λ 项 F , 都存在一 λ 项 M , 使得 $FM = M$ 成立, 这就是 λ 演算的不动点定理。

可以证明 λ 演算具有立奇-罗瑟性质。该性质说: 如果 λ 项 P 可以 β-归约到 M 和 N , 则必存在一 λ 项 T , 使 M 和 N 分别 β-归约到 T 。



这就是 β-归约的立奇-罗瑟性质。

并非每个 λ 项都具有 β-典范式, 例如 $(\lambda x. xx)$ ($\lambda x. xx$) 是一个 β-可约项, 对它进行一次 β-归约后得到 $[(\lambda x. xx)/x]xx$, 再用 $\lambda x. xx$ 分别替换项 xx 中的两个 x , 又得到 $(\lambda x. xx)(\lambda x. xx)$, 因此对这个 λ 项, β-归约可以无限长。1932 年 A. Church 证明判断任意一个 λ 项是否具有 β-典范式是一不可判定问题。

可以递归地将某些 λ 项赋予一个类型, 每个类型可以看成某些 λ 项组成的集合, 做法如下: 把变元 x 赋予类型 σ , 记为 $x: \sigma$; 若 λ 项 M 的类型为 η , 则 $\lambda x: \sigma. M$ 就是一带类型的 λ 抽象, 它的类型为 $\sigma \rightarrow \eta$ 。若 M, N 都是带类型 λ 项, 它们的类型分别为 $\sigma \rightarrow \eta$ 及 σ , 则 MN 也是一带类型的 λ 项, 它的类型为 η 。对带类型的 λ 项可以定义相应的 α-转换, β-归约, 并建立相应的归约理论, 带类型的 λ 演算也具有不动点定理, 立奇-罗瑟等性质。其中, 最重要的结果是强典范化性质, 即对任意的带类型 λ 项, 从它开始任意的归约序列都是有穷的, 从而带类型 λ 项是否有典范式问题是可判定的。这就是许多程序设计语言中变量、函数和过程都具有类型, 并由编译程序进行类型检查的理论基础。

前面关于 λ 项, α-转换, β-归约都是形式定义的, 因此 λ 演算可以定义成一个形式系统。能否找到一个数学理论作为 λ 演算形式系统的模型是 λ 演算模型论的主要研究问题。其难点是: 有些 λ 项, 如 $(\lambda x. x)(\lambda x. x)$ 是形式上合理的 β-归约项, 它可以 β-归约为 $(\lambda x. x)$, 但如果将 x 解释为某集合 σ

的元素,则 $\lambda x. x$ 就解释为定义域、值域均为 σ ,即类型为 $\sigma \rightarrow \sigma$ 的函数。这个函数在集合论里只能被定义域 σ 中的元素作用,而不能被 $\sigma \rightarrow \sigma$ 中的元素作用,因此 $(\lambda x. x)(\lambda x. x)$ 在经典集合论中就不能得到合理的解释。1969 年 D. Scott 建立了论域理论。其基本对象是完全偏序集(简记为 cpo)和建立在完全偏序集上的连续函数。D. Scott 首先证明,如果 D_0 是一任意 cpo,则以 D_0 为定义域和值域,类型为 $D_0 \rightarrow D_0$ 的全体连续函数也构成一个 cpo,记为 $D_1 \equiv [D_0 \rightarrow D_0]$,且存在 D_0 到 $[D_0 \rightarrow D_0]$ 的一一对应关系。顺此,令 $D_{n+1} \equiv [D_n \rightarrow D_n]$ 可以构造下述完全偏序集序列

$$D_0, D_1, \dots, D_n, D_{n+1}, \dots$$

D. Scott 进一步证明上述序列收敛到一完全偏序集 D_∞ ,且满足 $D_\infty \equiv [D_\infty \rightarrow D_\infty]$ 。这样 $(\lambda x. x)(\lambda x. x)$ 在 D_∞ 中就成为合法的连续函数了。使用 D. Scott 的模型论思想可以将程序设计语言中诸如 while true do skip end 这类死循环语句,解释为相应完全偏序集上的连续函数。从而为指称语义学奠定了理论基础。

参考文献

1. Barendregt H P. The lambda calculus, its syntax and semantics. North Holland, 1984
2. Hindley J R., Seldin J. Introduction to Combinators and λ -Calculus. Cambridge Press, 1981

(李未 王驹)

π yansuan

π 演算 (π -calculus) 基于通信系统演算 (CCS) 的可描述通信拓扑结构动态变化的分布式通信系统(例如移动电话系统)的传名演算。又称移动进程演算。 π 演算由 R. Milner, J. Parrow 和 D. Walker 三人提出。该演算允许进程之间传送和接收通道名,并且可以引入和输出局部名。因此, π 演算不仅非常简洁,而且表达能力很强。

π 演算的基本实体是通道名,通道名一方面可用于标识通道,另一方面可作为通道传送的消息。令 \mathcal{N} 是通道名的可数无穷集,其元素用 a, b, x, y, \dots 表示,令 t 表示进程, π 演算的语法可由以下 BNF 给出:

$$t ::= 0 \mid \alpha \cdot t \mid t + t \mid [x = y]t \mid t \mid t \mid (x)t \mid A(y_1, \dots, y_n)$$

$$\alpha ::= \tau \mid a(x) \mid \bar{a}x$$

π 演算的进程构造算子大部分来源于 CCS: 0

为不活跃进程, $\alpha \cdot t$ 为动作前缀, $+$ 为非确定选择, $[x = y]t$ 为等名测试, \mid 为并发合成, $(x)t$ 为限名,表示 x 是 t 的私有名。每个进程常量 A 都有一个定义方程 $A(x_1, \dots, x_n) \Leftarrow t$ 。

在 π 演算中有三类基本动作: τ 表示内部通信动作, $a(x)$ 表示输入动作, $\bar{a}x$ 表示输出动作。当 $a \neq x$ 时,可将 $(x)\bar{a}x \cdot t$ 缩写为 $\bar{a}(x) \cdot t$,导出动作 $\bar{a}(x)$ 称为约束输出动作。在 $a(x) \cdot t, (x)t$ 和 $\bar{a}(x) \cdot t$ 中, x 为具有辖域 t 的约束名。由此引出进程项 t 的约束名字集 $bn(t)$ 和自由名字集 $fn(t)$ 。

与传值 CCS 不同的是,由于 π 演算不区分数据变元和通道名,故限名算子 (x) 不仅能限制动作的主体——通道,而且能限制动作的客体——被传送的名字,如 $(x)\bar{a}x \cdot t$ 。因其私有性,被限制的私有名不能作为通道与外部通信。但能通过其他的通道向外界输出,进而将其辖域扩展到接收该私有名的进程。这一点是 π 演算与传值 CCS 的主要差异,也是导致 π 演算表达能力丰富和语义复杂的根源。因此在 π 演算中有两类含义不同的约束名:一类是形如 $(x)t$ 的私有名 x ,不能被实例化;另一类是形如 $a(y) \cdot t$ 的输入参数名 y , y 只是一个占位子,可被实例化。这两类约束名都可进行 α 换名。

π 演算的迁移语义如下所示:

$$\begin{aligned} \text{pre} & \frac{}{\alpha \cdot t \xrightarrow{\alpha} t} \\ \text{sum} & \frac{t \xrightarrow{\alpha} t'}{t + u \xrightarrow{\alpha} t'} \\ \text{match} & \frac{t \xrightarrow{\alpha} t'}{[x = x]t \xrightarrow{\alpha} t'} \\ \text{par} & \frac{t \xrightarrow{\alpha} t' \quad bn(\alpha) \cap fn(u) = 0}{t \mid u \xrightarrow{\alpha} t' \mid u} \\ \text{com} & \frac{t \xrightarrow{a(x)} t', u \xrightarrow{\bar{a}y} u'}{t \mid u \xrightarrow{\tau} [y/x]t' \mid u'} \\ \text{res} & \frac{t \xrightarrow{\alpha} t'}{(x)t \xrightarrow{\alpha} (x)t'} \quad x \notin n(\alpha) \\ \text{open} & \frac{t \xrightarrow{\bar{a}x} t'}{(x)t \xrightarrow{a(x)} t'} \quad x \neq a \\ \text{close} & \frac{t \xrightarrow{a(x)} t', u \xrightarrow{\bar{a}(x)} u'}{t \mid u \xrightarrow{\tau} (x)(t' \mid u')} \end{aligned}$$

$$\text{id} \quad \frac{t[\bar{y}/\bar{x}] \xrightarrow{\alpha} t'}{A(\bar{y}) \xrightarrow{\alpha} t'} \quad A(\bar{x}) \Leftarrow t$$

由于在 π 演算中抹杀了常量名和变量名、数据名和通道名之间的区别,为 π 演算引入基于互模拟概念的行为等价语义理论时将出现新的问题。直接根据强互模拟基本定义得到的关系只是一个等价关系,而不是同余关系,它不能被名字替换所保持,因而也不能被输入动作所保持。

将类似于传值 CCS 的这种基本的互模拟关系称为基互模拟。很自然地, π 演算的互模拟同余关系(简称为互模拟同余)“ \sim ”可定义为能被所有名字替换保持的基互模拟关系,即, $t \sim u$ 当且仅当对任意名字替换 σ , 均有 $t\sigma \sim u\sigma$ 。此时所有的自由名都看成变量。互模拟关系的这种二步定义法是 π 演算的一个有趣的现象和重要的特色。当然,与传值 CCS 类似,根据对输入动作的参数名的不同实例化策略, π 演算进程之间的基互模拟等价和相应的互模拟同余均有迟、早之分。

在 π 演算中,名字用来引用对象,一个进程所具有的自由名代表了该进程当前所拥有的关于其他进程的知识或与其他进程的联系。所有其他对象都对等地视为进程,通过原子事件进行交互。利用所谓“分子动作”技术,所有复杂数据类型以及基本函数,都可通过编码定义为进程,而不作为基本对象。 π 演算进程间所能传送的对象只能是通道名,而不能为进程,即只能用传递引用代替传递进程。虽然 π 演算是一阶的,但利用通道名作为指向进程的指针,容易将高阶进程翻译到 π 演算中去。

π 演算是对应用的广度(理论的一般性)和理论的完美同时追求的结果,通过较高抽象级来获得二者的统一。就概念简洁性和一般性而言, π 演算类似于作为函数计算基础的 λ 演算,因而可作为研究并发通信系统的基本工具。

参考文献

1. Milner R, Parrow J and Walker D. A calculus of mobile processes, Parts I and II. Journal of Information and Computation, 1992, 100: 1 ~ 77
2. Bergstra J A, Ponse A and Smolka S A, Editors. Handbook of Process Algebra. Elsevier, 2001

(李舟军)

ω -youxian zidongji

ω -有限自动机 (ω -finite state automaton)

一种在无限串上运行的有限状态自动机,是一种 ω -语言的识别模型。主要研究 ω -的各种识别方式以及在通常的五种识别条件下,识别的 ω -语言族之间的关系。特别,通过其中一种条件(即所谓 C_5)下识别的 ω -语言定义了 ω -正则语言,这是一种使 ω -自动机识别能力最强的识别方式。 ω -自动机理论的核心课题之一,是对 ω -正则语言的研究,包括对 ω -正则语言的描述及其性质的研究。

ω -自动机最早在文献中出现的是 J. R. Buchi (1960) 利用工作在无限序列上的有限自动机获得关于受限二阶逻辑理论的一个判定过程。自此以后一些研究 ω -自动机的各种形式体系的论文陆续出现,其中 J. R. Buchi, (1965, 1969), C. C. Elgot 和 M. O. Rabin (1966, 1969) 等人的论文均受到这些模型与二阶逻辑理论之间的密切关心的启发,因此重点放在判定问题。D. E. Muller (1963) 利用确定的 ω -有限自动机研究异步开关理论中的某些问题。R. McNaughton (1966) 首先发展了被 ω -有限自动机识别的 ω -语言的理论,即所谓的 ω -正则语言的理论。

ω -有限自动机研究的内容包括 ω -有限自动机的定义,五种识别条件, ω -正则语言的概念,对 ω -正则语言的描述以及与五种识别模型相应的五个 ω -语言族之间的关系。

ω -串与 ω -语言 设 Σ 是有限字母表,由 Σ 中的字母组成的无限序列,称为 Σ 上的 ω -串。用 Σ^ω 表示 Σ 上的所有 ω -串的集合。 Σ^ω 的任意子集称为 Σ 上的 ω -语言。

ω -有限自动机 一个五元组 $M = (K, \Sigma, \delta, q_0, F)$, 其中 K 为状态有限集, Σ 为输入字母表, $\delta: K \times \Sigma \rightarrow 2^K$, $q_0 (\in K)$ 为初始状态, $F (\subseteq 2^K)$ 为指定状态集族。如果 $\delta: K \times \Sigma \rightarrow K$, 则 M 是确定的 ω -有限自动机。

设 $\sigma = a_1 a_2 \cdots a_n \cdots$, $a_i \in \Sigma$, $i = 1, 2, \cdots$ 。状态序列 $r = \{q_i\}$, 称为 M 在 σ 上的一个运行,当且仅当 $q_i \in \delta(q_{i-1}, a_i)$, $i = 1, 2, \cdots$ 。一个运行确定一个映射 $f_r: N \rightarrow K$, $f_r(i) = q_{i-1}$, $i = 1, 2, \cdots$ 。令 $I(r) = \{q \in K \mid \text{card}(f_r^{-1}(q)) \geq \omega\}$, $O(r) = \{q \in K \mid f_r^{-1}(q) \neq \emptyset\}$ 。

ω -有限自动机的识别条件 包括 C_1, C_2, C_3, C_4 与 C_5 五个条件。 ω -有限自动机 M 在 C_i 条件下识别 ω -串 σ , 当且仅当存在 M 在 σ 上的一个运行 r , 使满足 C_i , $i = 1, 2, 3, 4, 5$ 。其中

C_1 : 存在 $H \in F$, 使 $I(r) \cap H \neq \emptyset$

C_2 : 存在 $H \in F$, 使 $I(r) \subseteq H$

C_3 : 存在 $H \in F$, 使 $O(r) \cap H \neq \emptyset$

C_4 : 存在 $H \in F$, 使 $O(r) \subseteq H$

C_5 : 存在 $H \in F$, 使 $I(r) = H$

设 $M = (K, \Sigma, \delta, q_0, F)$ 是一 ω -有限自动机, 称集合

$$T_i(M) = \{\sigma \in \Sigma^\omega \mid \text{存在 } M \text{ 在 } \sigma \text{ 上的一个运行 } r, \text{ 使满足 } C_i\}$$

为 M 在 C_i 条件下识别的 ω -语言, $i=1, 2, 3, 4, 5$ 。

ω -正则语言 Σ 上的一个 ω -语言 L , 如果存在一个 ω -有限自动机 M , 使 $T_5(M) = L$, 则称 L 为一个 ω -正则语言。R. Hosseley 证明了在上述五种识别模型中, 在 C_5 条件下 ω -有限自动机的识别能力最强。因此, 在 C_5 条件下的识别常常简单地说是“识别”, 而用 $T(M)$ 表示被 ω -有限自动机 M 识别的 ω -语言, 即

$$T(M) = \{\sigma \in \Sigma^\omega \mid \text{存在 } M \text{ 在 } \sigma \text{ 上的一个运行 } r, \text{ 使 } I(r) \in F\}.$$

具有单指定集的 ω -有限自动机 一种指定集族为单一集合的 ω -有限自动机, 称为单指定集的 ω -有限自动机。容易证明, 对于 C_1, C_3 , 单指定集的 ω -有限自动机与一般的 ω -有限自动机等价。

ω -克林尼闭包 一种描述 ω -正则语言特征的工具。令 \mathcal{L} 表示正则集族, ω -语言族

$$\omega-KC(\mathcal{L}) = \{L \subseteq \Sigma^\omega \mid L = \bigcup_{i=1}^K U_i V_i^\omega, U_i, V_i \in \mathcal{L}, i = 1, \dots, K, K = 1, 2, \dots\}$$

称为 \mathcal{L} 的 ω -克林尼闭包

ω -正则语言的特征定理 对 Σ 上的 ω -语言 L , 下列四个结论等价:

- (1) L 属于 $\omega-KC(\mathcal{L})$;
- (2) 存在一个识别 L 的 ω -有限自动机;
- (3) 存在一个在 C_1 条件下识别 L 的单指定集的 ω -有限自动机;
- (4) 存在一个识别 L 的确定的 ω -有限自动机。

参考文献

Rina S Cohen and Arie Y. Gold. Theory of ω -Languages I: Characterization of ω -Context-Free Languages. J. Comput. System Sci. 1977(15), 169 ~ 184

(苏锦祥)

Σ (ji diao) dai shu

Σ (基调)代数 (Σ (signature) algebra) 一种非齐性代数, 由 G. Birkhoff 等于 1970 年作为对齐

性代数的推广而引进。在非齐性代数中, 元素集被分成几个互不相交的子集。每个代数运算均以特定的子集为其定义域和值域。描述这种结构的语法称为基调。

基调和 Σ 代数 令 $S = \{s_i \mid i \in I\}$ 为一有限集, 其中 I 是一个有限下标集, 每个 s_i 称为一个类子(可以看作 Σ 代数中元素的类型)。 $O = \{O_j \mid j \in J\}$ 为另一有限集, J 也是有限下标集, 每个 O_j 称为一个运算。一个 k 目运算 $O_j (k \geq 0)$ 可表为

$$O_j: s_1 \times s_2 \times \dots \times s_k \rightarrow s_{k+1} \quad (1)$$

其中 $s_1, \dots, s_k, s_{k+1} \in S$ 。对偶 $\Sigma = \langle S, O \rangle$ 称为一个基调。

给定基调 $\Sigma_0 = \langle S, O \rangle$ 。假定有一组集合 $A = \{A_i \mid i \in I\}$ 和一组函数 $G = \{f_j \mid j \in J\}$, 使得诸类子 s_i 和诸集合 A_i 之间有一一对应关系, 诸运算 O_j 和诸函数 f_j 之间也有一一对应关系, 且 $\forall i \neq j, A_i \cap A_j = \emptyset$ 。若函数 f_j 与式 (1) 的 O_j 相对应, 则 $f_j: A_1 \times A_2 \times \dots \times A_k \rightarrow A_{k+1}$ 。满足这些条件的对偶 $\langle A, G \rangle$ 称为一个以 Σ_0 为基调的 Σ 代数(或基调代数), A 是它的载体集。

例如, 为了定义基调“整数堆”, 可设立 3 种类子: $s_1 = \text{int}, s_2 = \text{bag}, s_3 = \text{bool}$, 和 9 个运算:

empty: $\longrightarrow \text{bag}$
 zero: $\longrightarrow \text{int}$
 true: $\longrightarrow \text{bool}$
 false: $\longrightarrow \text{bool}$
 suc: $\text{int} \longrightarrow \text{int}$
 pred: $\text{int} \longrightarrow \text{int}$
 insert: $\text{bag} \times \text{int} \longrightarrow \text{bag}$
 remove: $\text{bag} \times \text{int} \longrightarrow \text{bag}$
 element: $\text{bag} \times \text{int} \longrightarrow \text{bool}$

令函数集 $G = \{\emptyset (\text{空堆}), 0, T, F, \text{eps}, +1, -1, \text{ins} (\text{向堆中插入元素}), \text{rem} (\text{从堆中删去元素}), ? (\text{判断某整数是否为堆中元素})\}$ 对应于上述运算集。又令载体集 $A = \{A_1, A_2, A_3\}$, $A_1 = \{0, 0+1, 0-1, 0+1-1, 0+1+1, \dots\}$, $A_2 = \{\emptyset, \text{ins}(\emptyset, 0), \text{rem}(\emptyset, 0), \text{ins}(\text{ins}(\emptyset, 0+1), 0), \dots\}$, $A_3 = \{T, F, ?(\emptyset, 0), ?(\text{rem}(\emptyset, 0), 0+1), \dots\}$ 。则 $\bar{A} = \langle A, G \rangle$ 是对应于上述基调“整数堆”的一个 Σ 代数。

Σ 代数的层次结构 设 $\langle A, G \rangle$ 和 $\langle A', G' \rangle$ 是对应于同一基调的两个 Σ 代数。如果存在单值映射 φ , 把 A 映射为 A' , G 映射为 G' , 且

- (1) $\varphi = \{\varphi_i \mid i \in I\} \cup \{\varphi_G\}$;
- (2) 对 $a_i \in A_i, \varphi_i(a_i) \in A'_i$, 其中 A_i 和 A'_i 分别

是 A 和 A' 中的载体;

(3) 对 Σ 中任一运算 O_j , 若 f 是 G 中与 O_j 对应的函数, 则 $\varphi_c(f) = f'$ 是 G' 中与 O_j 对应的函数;

(4) 由 $f(a_1, a_2, \dots, a_k) = a_{k+1}$, 其中诸 a_i 属于 A_i , 可得

$$\varphi_{k+1}(f(a_1, \dots, a_k)) = f'(\varphi_1(a_1), \dots, \varphi_k(a_k))$$

则 φ 称为是 $\langle A, G \rangle$ 到 $\langle A', G' \rangle$ 的一个 Σ 同态。 Σ 同态使 Σ 代数的类子属性不变, 也使它的函数结构不变。如果把 Σ 代数看作对象, 把 Σ 同态看作对象间的态射, 则对应于同一基调的所有 Σ 代数及其 Σ 同态构成一个范畴。

如果 Σ 代数 Σ_i 属于以某个 Σ 为基调的范畴 C , 使得对 C 中的每个 Σ 代数 Σ_i 都存在一个惟一的 Σ 同态 $\varphi_i: \Sigma_i \rightarrow \Sigma_i$, 则称 Σ_i 为 C 中的初始代数。如果存在 C 中的另一个 Σ 代数 Σ_2 , 使得对 C 中的每个 Σ_j , 都存在一个惟一的 Σ 同态 $\varphi_j: \Sigma_j \rightarrow \Sigma_2$; 则称 Σ_2 为 C 中的终结代数。初始代数和终结代数在同构意义下都是惟一的。在上例中, \bar{A} 是整数堆的初始代数。若令 $A'_1 = \{0\}$, $A'_2 = [\emptyset]$, $A'_3 = \{T(=F)\}$, $A' = \{A'_1, A'_2, A'_3\}$, 则 $\bar{A} = \langle A', G \rangle$ 也是对应于整数堆的一个 Σ 代数, 而且是它的终结代数。

项代数和有限生成代数 任意基调 Σ 的初始代数的存在性可通过它的基项代数 $T(\Sigma)$ 的存在性来证明。 $T(\Sigma)$ 定义如下:

(1) 对应于运算集 O 中的每个零目运算 O_j , $O_j: \longrightarrow s_i$ 有一个常量 O_j 属于 A_i ;

(2) 若 a_1, a_2, \dots, a_k 分别属于 A_1, A_2, \dots, A_k , 又有运算 $O_h \in O$, $O_h: s_1 \times s_2 \times \dots \times s_k \longrightarrow s_{k+1}$, 则有 $O_h(a_1, a_2, \dots, a_k) \in A_{k+1}$;

(3) 除此之外, 各 A_i 无其他元素;

(4) 令 $A = \{A_i\}$, 则 $T(\Sigma) = \langle A, O \rangle$;

(5) 每个 A_i 的每个元素称为一个基项。

容易证明 $T(\Sigma)$ 的存在性, 以及 $T(\Sigma)$ 是初始代数。在上面的例子中, $T(\text{整数堆})$ 和 \bar{A} 同构。

一个 Σ 代数称为是有限生成的, 如果它的每个元素都是用有限多个函数符号构造而成, 并且不含其他符号。基项代数是有限生成代数, 它的所有满 Σ 同态映象构成一个完全格。

抽象数据类型 在基项代数中允许每个载体 A_i 包含有限多个变量, 经函数集 O 作用后得到含变量的项, 如此生成的 Σ 代数称为项代数。一个 Σ 等式 e (又称公理) 写成 $t_1 = t_2$ 的形式 (可以推广为更复杂的形式), 其中 t_1 和 t_2 都是项代数中的元素。对偶 $D = \langle \Sigma, E \rangle$ 称为一个抽象数据类型, 其中 Σ 是

一个基调, E 是一组 Σ 等式。

Σ 等式组 E 在基调 Σ 的每个 Σ 代数 \bar{A} 上定义一个同余关系。即: 它首先是一个等价关系。其次, 对每个 $f \in G$, 必能从 a_i 等价于 b_i ($1 \leq i \leq n$) 中导出 $f(a_1, \dots, a_n)$ 等价于 $f(b_1, \dots, b_n)$ 。利用 E 定义的同余关系可求出 \bar{A} 的商代数 \bar{A}/E , 它仍然是一个 Σ 代数, 称为 D 的一个模型。 D 的全体模型及其 Σ 同态构成一个范畴, 模型 $T(\Sigma)/E$ 是此范畴中的初始代数, 也称为 D 的初始模型。它表达了 D 的初始语义。 D 还可以有其他的语义, 如终结语义。

偏 Σ 代数 把用某种程序设计语言写的程序看成一个抽象数据类型, 就可以用抽象数据类型的语义来描述程序的语义, 称为代数语义。对于只有表达式计算的函数式语言, 可令语言中的类型和函数分别对应于抽象数据类型中的类子和运算, 并用抽象数据类型中的公理来描述类型和函数的计算规则。对于含有语句的命令式语言, 可把程序的状态 (各变量在每一时刻的值) 作为新的类子, 而语句则是作用于状态类子之上的新的运算。对于含循环和 go to 语句等动态结构的语言, 可把程序的执行历史, 程序执行的当前位置等动态信息作为新的类子, 而把引起动态信息改变的语句结构作为新的运算。

为了表达程序中出现的非正常情况 (如运算无定义、运行不终止), 在基调中引进专用函数 defined。一个 Σ 等式可取下列 3 种形式:

(1) $\text{defined}(t) = \text{true}$;

(2) $\text{defined}(t) = \text{false}$;

(3) $t_1 = t_2$ (隐含 $\text{defined}(t_1) = \text{true}; \text{defined}(t_2) = \text{true}$)。

其语义是: 若 $\text{defined}(t) = \text{true}$ 在公理系统 E 中是可证的, 则项 t 在该抽象数据类型 D 的任何模型中都有定义。若 $\text{defined}(t) = \text{false}$ 是可证的, 则 t 在任何模型中均无定义。若两者都不能证明, 则 t 可以在有些模型中有定义, 而在另一些模型中无定义。这种抽象数据类型称为偏抽象数据类型, 相应的 Σ 代数称为偏 Σ 代数。例如, 在抽象数据类型“整数”中, 无穷大元素 ∞ 在标准模型中无定义, 而在非标准模型中有定义。

Σ 同态 φ 称为是强的, 若 $\varphi(t)$ 有定义当且仅当 t 有定义。 φ 称为是弱的, 若由 t 无定义可知 $\varphi(t)$ 一定无定义。 φ 称为是共弱的, 若由 t 有定义可知 $\varphi(t)$ 一定有定义。相应地可定义弱和共弱的初始模型和终结模型。已经证明, 任何一个偏抽象

数据类型 D 都有一个共弱初始模型和弱终结模型。如果 D 是描述程序语义的,那么此弱终结模型与该程序的最小不动点语义等价。

参考文献

陆汝钊. 计算机语言的形式语义. 北京: 科学出版社, 1992 (陆汝钊)

lingxing wenfa

0 型文法 (type 0 grammar) 参见短语结构文法。

yixing wenfa

1 型文法 (type 1 grammar) 参见上下文有关文法。

erxing wenfa

2 型文法 (type 2 grammar) 参见上下文无关文法。

sanxing wenfa

3 型文法 (type 3 grammar) 参见正则文法。

条目汉语音序索引

说 明

1. 本索引供读者按条题名的汉语拼音寻查条目。为了保证全书条目的完整性,外文条题名的条目也收入本索引内。
2. 本索引按条题名的汉语拼音字母顺序排列。第一字同音时,按四声顺序排列;同音同调时按笔画多少和笔顺排列;如完全相同,则按第二字,余类推。
3. 外文条题名排在最后,依次按英文字母、希腊字母和阿拉伯数字顺序排列。

A

阿克曼函数	1
安全服务	1
安全机制	1

B

巴克斯范式	3
巴克斯-诺尔形式体系	3
办公自动化	3
半导体存储器	4
半导体存储器芯片	6
半导体读写存储器	7
半结构化数据	9
绑定	9
笔记本电脑	10
边界网关协议	11
编辑程序	12
编码工具	12
编译程序	12
编译程序的编译程序	13
变量	14
变形体动画	14
标准带	14
标准通用置标语言	15
表处理语言	15
表达式	16
表面安装技术	16

表情动画	17
并发程序设计	17
并发程序设计语言	18
并发控制	18
并发模型	19
并行编译程序	20
并行程序设计	21
并行程序设计语言	22
并行处理系统	22
并行仿真	25
并行工程	26
并行数据库	27
并行算法	27
波分多路复用	30
波峰焊	30
波兰记法	31
波斯特对应问题	32
波斯特机	32
玻耳兹曼机	33
博弈树搜索	34
不间断电源	35
不可判定问题	37
布尔代数	37
布图技术	38
布线系统标准	39

C

参数估计	40
------------	----

串处理语言	82
串行传输	83
窗口系统	83
词法分析	84
磁存储器	84
磁带存储器	85
磁光盘驱动器	86
磁卡机	87
磁盘存储器	88
磁盘阵列	89
存储保护	91
存储管理	92
存储管理程序	93
存储器差错校验	94
存储器管理部件	94
存储器类型	95
存储器性能	96
存储器组成	96
存储区域网	96
存储系统	98
存储转发交换	99

打印机测试	101
大规模并行处理	101
大型计算机	104
大型计算机电源系统	105
代码生成	106
代码优化	107
代数规约	107
代数语义	108
单回路数字控制器	109
单片计算机	110
等值面构造技术	111
低轨道卫星通信系统	112
低级语言	113
地理信息系统	113
递归函数	114
点对点连接协议	115
电磁兼容性	115
电荷耦合器件存储器	117
电可擦编程只读存储器芯片	117
电缆接入技术	118
电路交换	118

电信管理网络	119
电源测试	119
电源集成电路	120
电子词典	121
电子计算机	121
电子商务	124
电子商务标准	124
电子设计自动化	126
电子数据交换	128
电子印前处理	129
电子邮件	130
电子政务	131
定理机器证明	131
定性推理	133
动画后期处理	133
动画描述语言	134
动态随机存取存储器芯片	134
短语结构文法	137
短语结构语法	137
对象	138
对象-关系数据库	138
多处理机系统总线	139
多道程序设计	141
多发射结构	141
多分辨率造型	142
多类代数	143
多路复用	144
多媒体技术	144
多媒体数据库	146
多媒体文档	146
多媒体文档规范	147
多媒体著作工具	148
多模态人机交互	148
多模态生物特征融合	149
多态类型	150
多项式空间归约	151
多项式谱系	151
多项式时间归约	152
多芯片模块	153
多值逻辑	154

E

二进制算术运算	155
---------------	-----

F

反传学习	159
范畴论	159
范式	161
防火墙	161
防信息泄漏技术	162
仿真语言	163
访问控制	164
非传统计算机	165
非单调逻辑	167
非单调推理	168
非过程语言	171
非击打式印刷机	171
非监督学习	173
非线性代数方程组数值解法	174
非真实感图形绘制	175
分布交互式仿真	176
分布式操作系统	177
分布式程序设计	177
分布式处理系统	178
分布式多媒体系统	179
分布式共享存储	180
分布式计算环境	180
分布式软件系统	181
分布式数据库	182
分布式数据库系统	182
分布式网络管理	183
分布式异构型计算机系统	183
分布式专家系统	184
分片	186
分时处理	186
分时共享模式	187
分析学习	187
分组交换	188
分组装拆器	190
封装	191
服务器	191
浮点数标准	192
辐射度技术	193
辅助存储器	194
附网存储	194
复杂性度量	196
复杂性归约	197

复杂指令集计算机 198

G

概率算法 199
 概率自动机 199
 干线子系统 202
 感知机 203
 高次代数方程解法 204
 高级语言 205
 高阶逻辑 205
 高速缓冲存储器 206
 高速缓冲存储器一致性 208
 高速局域网 209
 高速数字信号传输 210
 高性能计算 212
 哥德尔配数 213
 哥德尔完全性定理 214
 格 214
 格雷贝奇范式 215
 格语法 215
 个人数字助理 216
 个人通信网 216
 跟踪球 217
 工程数据库 217
 工具箱 218
 工业控制计算机 219
 工作区子系统 219
 工作站 220
 公告板系统 221
 公共管理信息协议 221
 公理语义 222
 公用交换电话网 223
 公用数据网 223
 公钥基础设施 224
 公钥密码技术 224
 功能规约 225
 功能规约语言 225
 功能语言 225
 供应过程 227
 共享存储 228
 构造类型 229
 固态硬盘 230
 故障恢复 233
 关键帧动画 233

关系 233
 关系代数 234
 关系数据库 235
 管理过程 236
 管理信息库 236
 管理信息系统 237
 管理子系统 238
 光笔 239
 光存储器 239
 光电集成电路 240
 光计算机 240
 光记录 241
 光记录编码方法 241
 光缆 242
 光亮度计算 243
 光盘镜像服务器 243
 光盘控制器 244
 光盘库 245
 光盘塔 245
 光纤到路边 246
 光纤分布数据接口 246
 光线跟踪技术 247
 光学标记阅读机 248
 光学字符阅读机 249
 光照模型 249
 广播路由选择 250
 广谱语言 250
 广域网 250
 广域信息服务系统 251
 归结方法 251
 归纳推理 252
 归纳学习 253
 归纳综合方法 255
 归约机 255
 规范化 256
 过程表示 257
 过程控制计算机 257
 过程实现方法 259
 过程式程序设计 260
 过程语言 261

H

函数式程序设计 262
 函数式程序设计语言 262

函数与过程	263
汉语拼音输入法	263
汉语言语(语音)合成	264
汉语言语(语音)理解	264
汉语言语(语音)识别	265
汉语言语(语音)识别分类	267
汉字	267
汉字笔画输入法	268
汉字编码(键盘)输入方法	269
汉字编码字符集	269
汉字部件(字根)输入法	271
汉字键盘输入	271
汉字内码扩展规范—GBK	272
汉字识别	273
汉字识别后处理	274
汉字识别基本方法	275
汉字识别特征	275
汉字形音输入法	276
汉字音形输入法	276
汉字字汇	277
汉字字型	277
汉字字序	278
盒带驱动器	278
宏处理程序	280
虹膜识别	280
互联网	281
环	285
回归分析	285
汇编程序	286
汇编语言	287
会合	287
绘图机	288
混合光纤同轴电缆	290
混合计算机	290
混合计算模型	291
混合自动机	292
获取过程	292
霍恩逻辑	293

J

击打式打印机	295
机器翻译	296
机器翻译评价	298
机器人传感器	299

机器人控制	300
机器人视觉	300
机器人运动规划	301
机器学习	302
机器语言	305
机器周期	305
机助人译	306
基数	306
基于 Agent 的计算	307
基于案例的推理	308
基于构件的软件开发方法	309
基于合一的语法理论	310
基于类型理论的方法	310
基于内容的检索	311
基于全视函数的绘制	311
基于图像的绘制	312
基于图像的造型	313
基于物理的动画	313
基于物理的造型	314
基准程序	314
激光照排机	315
集成电路	315
集成电路测试程序	319
集成电路测试系统	321
集成电路制造	323
集成数字环载波	327
集合	328
集合论	328
集合运算	330
集群式计算	331
集散控制系统	332
集线器	334
集中器	334
集中式网络管理	334
几何造型	335
计量语言学	336
计时器	336
计算复杂性理论	336
计算机 RAS 技术	339
计算机安全	341
计算机病毒	342
计算机簇	342
计算机代数	343
计算机电源	344

- | | | | |
|-------------------------|-----|----------------------------|-----|
| 计算机动画 | 345 | 计算数论 | 412 |
| 计算机仿真 | 346 | 计算语言学 | 414 |
| 计算机辅助工艺规划 | 348 | 记录类型 | 414 |
| 计算机辅助绘图 | 349 | 继承 | 414 |
| 计算机辅助技术 | 349 | 加法器 | 415 |
| 计算机辅助教学 | 351 | 加固技术 | 416 |
| 计算机辅助软件工程 | 352 | 假设检验 | 416 |
| 计算机辅助设计 | 352 | 假脱机 | 418 |
| 计算机辅助制造 | 353 | 剪裁过程 | 418 |
| 计算机辅助质量控制 | 354 | 简单类型 | 419 |
| 计算机故障诊断 | 355 | 简单网络管理协议 | 419 |
| 计算机过程控制 | 357 | 建筑群子系统 | 420 |
| 计算机过程控制方式 | 358 | 鉴别 | 421 |
| 计算机机房设施 | 359 | 键码 | 422 |
| 计算机集成制造系统 | 360 | 键盘 | 423 |
| 计算机可维护性 | 361 | 交叉编译程序 | 423 |
| 计算机可用性 | 362 | 交互式电视 | 424 |
| 计算机控制 | 363 | 交互式语言 | 424 |
| 计算机类型 | 365 | 交换机 | 425 |
| 计算机流水线 | 367 | 交换技术 | 426 |
| 计算机软件 | 369 | 交换式多兆位数据业务 | 426 |
| 计算机软件的法律保护 | 372 | 接口定义语言 | 427 |
| 计算机视觉 | 374 | 结构化布线系统 | 428 |
| 计算机视觉中的结构光法 | 375 | 结构化程序设计 | 429 |
| 计算机数学 | 376 | 结构化方法 | 429 |
| 计算机图形标准 | 378 | 结构化分析与设计技术 | 430 |
| 计算机图形学 | 379 | 解释程序 | 431 |
| 计算机网络 | 380 | 解释机制 | 431 |
| 计算机维护 | 385 | 界面工具 | 432 |
| 计算机系统可靠性 | 387 | 进程 | 432 |
| 计算机系统性能模拟 | 388 | 进程代数 | 433 |
| 计算机信息系统 | 390 | 精简指令集计算机 | 433 |
| 计算机信息系统安全保护等级划分准则 | 391 | 静态随机存取存储器芯片 | 435 |
| 计算机性能评价 | 391 | 静止图像压缩编码标准 | 437 |
| 计算机音乐 | 394 | 局部搜索算法 | 437 |
| 计算机应用技术 | 395 | 局域网 | 437 |
| 计算机硬件 | 396 | 局域网基准(参考)模型 | 439 |
| 计算机硬件可靠性 | 400 | 局域网介质访问控制方法 | 439 |
| 计算机游戏 | 401 | 局域网介质访问控制子层 | 440 |
| 计算机整机检测 | 402 | 局域网逻辑链路控制子层 | 440 |
| 计算机支持的协同工作 | 404 | 局域网拓扑结构 | 441 |
| 计算机组成 | 406 | 局域网物理介质无关子层 | 442 |
| 计算几何 | 409 | 局域网物理介质相关子层 | 443 |
| 计算理论 | 410 | 局域网协议标准(IEEE 802 标准) | 443 |

矩阵计算	445
矩阵特征值问题数值解法	447
句法模式识别方法	448
巨型计算机	449
距离图像获取与分析	451
决策支持系统	452
军事指挥信息系统	454

K

卡通动画	456
开发过程	456
开放系统	458
开放系统互连基准(参考)模型	461
开盘式磁带驱动器	463
抗恶劣环境计算机	465
科学计算可视化	465
可编程控制器	466
可编程只读存储器芯片	467
可擦编程只读存储器芯片	468
可穿戴计算	469
可计算函数	469
可计算性理论	470
可靠性设计	473
可扩充语言	474
可扩展置标语言	474
可判定问题	475
可视编程语言	475
可视程序设计	476
可视电话	477
可信计算机系统评估准则	477
客户-服务器计算	478
客户-服务器模式	479
课件	479
空间复杂性	480
空间数据库	481
控制杆	481
快可擦编程只读存储器芯片	482
快速傅里叶变换	483
快速以太网	484
宽带接入技术	486
宽带接入体系结构	486
宽带接入网	487
框架表示	487

L

类	489
类比推理	489
类比学习	490
类型定义	492
类型论	492
离散事件系统仿真	493
离散事件系统仿真建模方法学	495
离散事件系统仿真输出分析	497
离散数学	498
理解工具	498
立体视觉	499
例程	499
连接编辑程序	500
连接专家系统	500
连续系统仿真	502
联机分析处理	503
联机事务处理	505
联机手写汉字识别	505
联想存储器	506
联想记忆	506
量子计算	507
领域工程	508
浏览器	509
浏览器-万维网-数据库模式	509
流控	510
流媒体	510
流式磁带驱动器	511
路由器	511
路由选择	512
绿色计算机	513
论域理论	513
逻辑表示	514
逻辑程序设计	515
逻辑程序设计语言	516
逻辑推理机	517
逻辑运算	517
螺旋模型	518

M

马丁洛夫类型理论	520
媒体处理器	520
蒙特卡罗法	522

密码学 523
 面向对象程序设计 524
 面向对象方法 524
 面向对象数据库 525
 面向对象语言 526
 面向数据结构方法 526
 面向问题语言 527
 命令式语言 528
 命令语言 528
 命题逻辑 529
 模糊逻辑 530
 模糊推理 531
 模块化方法 533
 模块结构图 534
 模拟计算机 535
 模式 536
 模式分类器 536
 模式识别 537
 模数转换器 539
 模态逻辑 539
 模型检验 541
 模型论 542

N

内部路由协议 544
 内联网 544
 能力成熟度模型 545

P

排版软件 547
 排队论 548
 排序算法 548
 佩特里网论 549
 配置管理工具 552
 喷泉模型 552
 批处理 553
 片上系统 553
 偏微分方程数值解法 553
 频分多路复用 557
 平方逼近 558
 平均故障间隔时间 559
 平面图 559
 剖分曲面 560
 瀑布模型 560

Q

企业流程重组 562
 企业资源规划 562
 启发式搜索 563
 千兆位以太网 565
 前端处理器 566
 前后断言方法 566
 嵌入式操作系统 567
 嵌入式计算机 567
 乔姆斯基层次 568
 乔姆斯基范式 569
 切比雪夫逼近 569
 清洗盘 570
 曲面 570
 曲线 571
 全频带数字音频的编码 572
 全球定位系统 573
 全球移动通信系统 573
 全文检索 575
 全息照相存储器 575
 权标环网 576
 权标总线网 577
 群 577
 群件 578

R

绕接 580
 热设计 580
 人工神经网络 582
 人工神经网络在模式识别中的应用 584
 人工生命 586
 人工智能 586
 人机交互技术 588
 人机交互系统 589
 人脸识别 591
 人体动画 592
 人助机译 593
 任务调度程序 593
 容错计算 594
 容错计算机 595
 柔性制造系统 598
 软磁盘测试 599
 软磁盘存储器 600

软磁盘控制器	600	三维空间数据场可视化	638
软磁盘驱动器	601	三维网格处理	639
软磁盘驱动器测试	602	三维网格模型参数化	639
软磁盘适配器	603	三维网格模型简化	640
软件安全性	603	三维形态恢复	640
软件包	605	扫描仪	642
软件测试	605	筛法	642
软件调试	606	上下文无关文法	643
软件方法学	606	上下文有关文法	644
软件复用	608	设备测试	645
软件工程	609	设备间子系统	645
软件工程环境	612	设计工具	646
软件工程经济学	613	设计规约	646
软件工具	614	设计性语言	647
软件构件	615	摄像机标定	648
软件构件库	616	砷化镓存储器	648
软件过程	617	砷化镓集成电路	649
软件过程模型	618	神经计算	651
软件开发方法	619	神经计算机	651
软件开发环境	621	神经网络芯片	652
软件开发模型	622	甚小口径天线地球站	652
软件可靠性	623	生物计算	653
软件库	624	生物特征识别	654
软件理解	624	时段演算	656
软件流水	625	时分多路复用	657
软件逆向工程	626	时间复杂性	659
软件配置管理	626	时态逻辑	660
软件设计模式	627	时序系统	661
软件生存周期	627	实时绘制	662
软件体系结构	628	实体联系模型	662
软件维护	628	实体联系图	663
软件系统	629	事务处理	663
软件需求定义	630	事务进程监控器	663
软件语言	631	事务元	664
软件再工程	632	视觉计算理论	664
软件质量	632	视频光盘机	665
软件中间件	633	视频会议	666
软件主体	634	视频随机存取存储器芯片	667
软件自动化方法	635	视图插值	668
S		适应谐振理论	669
		收发器	670
三维动画	638	手势识别	670

输出设备	672	数据依赖	707
输入设备	672	数据终端设备	707
输入输出管理程序	673	数据准备设备	707
输入输出技术	674	数理逻辑	708
输入输出接口	674	数理统计	710
输入输出设备接口	674	数模转换器	711
输入输出通道	677	数值逼近	712
鼠标器	677	数值积分	713
属性文法	678	数值计算	715
树	679	数值计算误差分析	717
数据	679	数值微分	719
数据安全性	679	数制	720
数据编码	679	数字磁记录	722
数据仓库	681	数字磁记录编码方法	725
数据处理速率	682	数字磁记录检错码	728
数据传输	682	数字磁记录纠错码	729
数据传输差错检测与控制	684	数字多用途光盘	730
数据传送	686	数字化仪	731
数据电路设备	686	数字话音的压缩编码	732
数据调制	687	数字集成电路	732
数据共享	688	数字几何处理	735
数据结构	688	数字计算机	736
数据结构化系统开发方法	689	数字可写光盘	739
数据库	690	数字可重写光盘	740
数据库管理	691	数字签名	741
数据库管理系统	692	数字摄像头	741
数据库连通性标准	693	数字视频编辑	742
数据库设计	693	数字视频获取	742
数据库系统	694	数字数据网	743
数据库系统三级结构	695	数字图书馆	743
数据库性能评价	696	数字图像处理	744
数据类型	696	数字微分分析机	745
数据链路控制	696	数字系统逻辑综合技术	746
数据链路控制规程	697	数字系统模拟技术	747
数据流计算机	698	数字相机	747
数据流图	699	数字信号处理器	748
数据流语言	700	数字音频编辑	749
数据模型	700	数字音频获取	749
数据通路	701	数字用户专用线	750
数据通信	701	数组类型	751
数据通信接口	703	双绞线电缆	751
数据通信接口标准	704	水平子系统	752
数据挖掘	706	顺序程序设计	754
数据完整性	706	顺序控制	754

说话人识别	755
说明	756
说明性语言	757
私钥密码技术	757
死锁	758
素数	758
素性测试	759
算法	760
算法动画	761
算法设计	762
算法学	763
算术逻辑部件	764
算术逻辑运算	766
随机存取存储器芯片	766
随机存取机	767
孙子定理	768
索引	768

T

特征选择	770
体绘制技术	770
条码阅读器	771
调试工具	772
调整盘	772
调制解调器	774
停机问题	775
通信服务器	776
通信控制器	776
通信控制设备	776
通信顺序进程	777
通信系统演算	778
通用多八位编码字符集 ISO/IEC 10646	779
通用寄存器	780
同步传输	780
同步光纤网	781
同余	782
同轴电缆	782
统一建模语言	783
图灵归约	784
图灵机	784
图论	786
图论算法	787
图像边界表示	788
图像边缘检测	789

图像变换	790
图像变换运算	792
图像表示	792
图像并行处理	793
图像重建	794
图像处理的基本运算	795
图像的压缩编码	796
图像点运算	798
图像反向滤波复原	799
图像分割	800
图像分析	800
图像复原	800
图像骨架表示	801
图像获取	801
图像几何特征表示	802
图像几何运算	803
图像矩表示	804
图像理解系统	805
图像邻域运算	806
图像模型	806
图像区域表示	808
图像区域分割	809
图像特征提取	809
图像退化	811
图像维纳滤波复原	812
图像细化	813
图像像素分类	813
图像形态学运算	814
图像运动模糊复原	815
图像增强	816
图形变换	817
图形裁剪	817
图形反走样技术	817
图形元文件标准	818
图元生成	818
推理机	819
脱机打印设备	820
脱机设备	820
脱机手写汉字识别	820

W

外部路由协议	822
外存储设备接口	822
外存储子系统	824

外联网	826	微处理器	857
完全偏序	826	微控制器	858
万维网	826	微内核	860
万维网数据管理	827	微型计算机	860
网格计算	827	微组装技术	862
网关	830	维护工具	863
网际协议	830	维护过程	864
网络安全	831	维也纳开发方法	864
网络操作系统	832	伪随机数	865
网络测试	832	卫星接入技术	866
网络处理器	833	卫星通信	866
网络服务	834	文档语言	868
网络服务质量	835	文法	868
网络工程	836	文化程序设计	869
网络管理	836	文件	872
网络管理标准	837	文件传送	872
网络规划	838	文件共享	874
网络互连	839	文件管理程序	875
网络互连技术	839	文语转换	875
网络互连设备	840	纹理生成	876
网络互连协议	840	纹理映射	876
网络集成服务	841	吴方法	877
网络计算模式	841	无焊压接	878
网络区分服务	841	无线局域网	878
网络入侵检测	842	无线应用协议	880
网络软件	842	无向图	882
网络设计	843		
网络适配器	845	X	
网络数据单元	845	系统程序设计语言	883
网络数据库	846	系统兼容性	883
网络体系结构	847	系统维护	884
网络协议	848	系统性能指标	885
网络协议工程	849	系统总线	886
网络协议规范	850	细胞自动机	887
网络协议形式描述技术	850	下推自动机	888
网络协议一致性测试	851	显示器	889
网络应用服务	852	现场总线控制系统	892
网络运行环境	852	线程	893
网络中间件	853	线性代数方程组数值解法	894
网络资源预约协议	853	线性逻辑	897
网桥	854	线性文法	898
网状数据库	855	线性有界自动机	898
微波通信	856	相变光盘驱动器	899
微程序控制器	856	向量计算	900

项目管理工具	900
消息处理系统	901
消息传递	902
消隐技术	903
小脑网络模型	904
小型计算机	905
协处理器	905
协同例程	906
新一代网际协议	907
信念	908
信息	909
信息检索方法	909
信息检索中的相关反馈	909
信息交换用汉字编码字符集	
GB 2312—1980	910
信息交换用汉字编码字符集的扩充	
GB 18030	911
信息系统安全评估准则	912
信息隐蔽	912
信元交换	912
形式方法	912
形式功能规约	914
形式规约	914
形式设计规约	916
形式语言理论	916
形式语义	919
虚拟存储器	919
虚拟机	921
虚拟现实	922
虚拟现实输出设备	923
虚拟现实输入设备	924
虚拟终端	925
虚拟专网	926
需求定义语言	926
需求分析工具	928
需求工程	929
序数	929
学习计算理论	930
寻址方式	931
训练仿真器	932

Y

言语(语音)合成方法	934
言语(语音)合成器	934

言语(语音)识别的特征抽取	935
言语(语音)识别中的语言模型	936
颜色复制	936
颜色管理	937
颜色模型	938
演化模型	940
演绎数据库	941
演绎综合方法	941
样条函数	942
遥感信息处理	943
页面描述语言	944
一阶逻辑	945
移动式计算机	947
移动数据库	948
移动通信	949
移动通信网	949
遗传算法	949
遗传学习	952
以太网	953
异步传输	955
异步传送模式	955
异步传送模式局域网	957
异常处理	958
译后编辑	959
译前编辑	959
因子分解	959
隐函数曲面	959
印前图像处理技术	960
印刷体汉字识别	961
印刷文本版面分析	961
印制板测试	962
印制板设计	963
印制板在线测试	965
印制板制造	967
映射	968
硬磁盘存储器	968
硬磁盘控制器	970
硬磁盘驱动器	971
硬磁盘驱动器测试	972
硬磁盘适配器	973
硬件描述语言	973
硬件同步机制	974
硬件验证	976
硬连线控制器	976

用户界面 976
 用户界面管理系统 977
 用户数据报协议 978
 有限元方法 979
 有限自动机 981
 有向图 983
 语法 983
 语法分析 983
 语法图 984
 语境分析 984
 语句 985
 语料库 986
 语料库语言学 988
 语言处理系统 989
 语义 991
 语义分析 991
 语义网络表示 992
 语用 992
 域 992
 域名系统 993
 元器件测试 994
 元器件可靠性 995
 元器件老化 995
 元器件筛选 996
 元语言 996
 原始递归函数 997
 原型速成方法 997
 远程教育 998
 远程网络监控 999
 远程医疗 1000
 约束推理 1002
 运动分析 1003
 运动图像压缩编码标准 1003
 运算器 1005
 运算速度评价 1005
 运作过程 1006

Z

造型技术 1008
 增强现实 1008
 栈自动机 1009
 锗硅异质结器件 1009
 真实感图形生成 1011
 阵列处理机 1012

正则表达式 1012
 正则文法 1013
 帧中继交换 1015
 支持过程 1016
 知识表示 1017
 知识工程 1020
 知识获取 1021
 知识获取工具 1023
 知识精化 1025
 知识库 1026
 直接表示 1027
 直接存储器存取 1028
 直觉主义逻辑 1028
 直流电源 1029
 值 1032
 只读存储器 1032
 只读存储器芯片 1032
 只读光盘驱动器 1035
 指称语义 1036
 指挥控制通信情报系统 1037
 指令格式 1037
 指令级并行处理 1038
 指令寄存器 1039
 指令类型 1039
 指令系统 1041
 指纹识别 1042
 指针类型 1043
 智能机器人 1043
 智能卡阅读器 1045
 置标语言 1046
 中断 1047
 中国邮路问题 1047
 中继器 1048
 中日韩统一汉字 1049
 中文信息处理 1050
 中文信息检索 1050
 中央处理器 1051
 终端 1055
 终端服务器 1055
 终端设备 1055
 主存储器 1057
 主页 1057
 专家系统 1058
 专家系统开发环境 1060

专用逻辑集成电路	1061
专用小交换机	1063
转换检测缓冲器	1064
装入程序	1064
状态转移图	1064
资源共享模式	1065
子程序	1065
子例程	1066
字符集	1066
自编译程序	1067
自底向上方法	1068
自顶向下方法	1068
自动标引	1069
自动盒带库	1070
自动机理论	1071
自动推理	1072
自检验电路	1073
自然景物造型	1073
自然演绎法	1074
自然语言处理	1074
自然语言处理的词法分析	1075
自然语言处理的句法分析	1076
自然语言理解	1077
自组织映射模型	1078
综合理论性能	1079
综合业务数字网	1079
总线标准	1081
总线控制器	1082
总线仲裁器	1082
租用线路网	1083
组合逻辑	1084
组合算法	1084
组合学	1085
最大公因子	1088
最大流	1088
最短路径问题	1089
最弱前置条件方法	1089
最小二乘法	1089
最小生成树	1090
最优化方法	1091
作业	1093
作业管理程序	1093

A

A* 算法	1095
-------------	------

Ada 语言	1096
ALGOL 60 语言	1097
ALGOL 68 语言	1097
AO* 算法	1098

B

BASIC 语言	1099
BCY 语言	1100

C

C + + 语言	1100
CGI 图形接口标准	1101
CIP - L 语言	1101
COBOL 语言	1101
C 语言	1102

D

DOS 操作系统	1103
----------------	------

E

Eiffel 语言	1105
-----------------	------

F

FGSPEC 语言	1106
FORTRAN 语言	1107
FP 语言	1107

G

GKS 图形标准	1108
Gopher 服务	1108
Gouraud 明暗处理	1109
GSPEC 语言	1109

H

Hopfield 神经网络模型	1110
-----------------------	------

I

Internet	1111
Internet(因特网)服务提供者	1112
Internet 地址	1112
Internet 基本服务	1113
Internet 体系结构	1114

J

Jackson 系统开发方法	1115
----------------------	------

Java 语言 1116

K

Kerberos 鉴别 1117

L

Larch 语言 1117

LINPACK 基准程序 1118

Linux 操作系统 1118

LISP 语言 1119

Listserv 论坛 1120

LR(*k*) 文法 1120**M**

Miranda 语言 1120

ML 语言 1121

Modula-2 语言 1121

N

NP 类问题 1121

NP 完全问题 1122

NP 完全问题近似方法 1123

NP 完全性理论 1123

O

OBJ 语言 1125

Occam 语言 1125

OpenGL 图形标准 1126

OSI 管理体系结构 1126

OSPF 协议 1127

P

PARLOG 语言 1128

PASCAL 语言 1129

PDL 语言 1129

PHIGS 图形标准 1130

Phong 明暗处理 1131

PROLOG 语言 1131

PSA 程序 1131

PSL 语言 1132

P 类问题 1132

R

RIP 协议 1132

S

SETL 语言 1133

Smalltalk 语言 1133

SNOBOL 语言 1134

SPEC 基准程序 1135

T

TCP/IP 协议集 1135

TPC 基准程序 1137

U

Unicode 编码字符集 1137

UNIX 操作系统 1138

USENET 新闻 1139

V

VAL 语言 1140

VLIW 处理机 1140

VLSI 算法 1141

W

Windows 操作系统 1143

X

X.75 建议 1145

XCY 语言 1146

XYZ/E 语言族 1146

Z

Z 语言 1146

 α α - β 剪枝 1147 **λ** λ 演算 1148 **π** π 演算 1150

ω		1	
ω - 有限自动机	1151	1 型文法	1154
Σ		2	
Σ (基调) 代数	1152	2 型文法	1154
0		3	
0 型文法	1154	3 型文法	1154

INDEX OF ARTICLES

(条目外文索引)

说 明

本索引按条目的外文条题名的英文字母顺序排列。非英文字母的条题名则按希腊字母及阿拉伯数字的顺序排在后面。

A

- | | | | |
|--|------|---|------|
| A* algorithm | 1095 | anti-aliasing technique | 817 |
| abstract algebra | 74 | AO* algorithm | 1098 |
| abstract data type | 75 | application of artificial neural networks to | |
| access control | 164 | pattern recognition | 584 |
| Ackermann's function | 1 | application specific logic integrated circuit, ASIC | |
| acquisition of digital audio | 749 | | 1061 |
| acquisition of digital video | 742 | approximate method for NP complete problem | |
| acquisition process | 292 | | 1123 |
| Ada language | 1096 | approximation in quadratic norm | 558 |
| adaptive resonance theory, ART | 669 | arithmetic and logic unit, ALU | 764 |
| adder | 415 | arithmetic speed evaluation | 1005 |
| addressing mode | 931 | arithmetic unit | 1005 |
| administration subsystem | 238 | array processor | 1012 |
| agent-based computing | 307 | array type | 751 |
| algebraic semantics | 108 | artificial intelligence | 586 |
| algebraic specification | 107 | artificial life | 586 |
| ALGOL 60 language | 1097 | artificial neural networks | 582 |
| ALGOL 68 language | 1097 | assembler | 286 |
| algorithm | 760 | assembly language | 287 |
| algorithmic animation | 761 | associative memory | 506 |
| algorithmics | 763 | asynchronous transfer mode local area network, | |
| alpha-beta pruning | 1147 | ATMLAN | 957 |
| analog computer | 535 | asynchronous transfer mode, ATM | 955 |
| analogical reasoning | 489 | asynchronous transmission | 955 |
| analog-to-digital converter, ADC | 539 | attribute grammar | 678 |
| analytic learning | 187 | augmented reality, AR | 1008 |
| animation post-processing | 133 | authentication | 421 |
| animation script language | 134 | automated cartridge tape library | 1070 |
| | | automated reasoning | 1072 |
| | | automatic indexing | 1069 |

automaton theory	1071
auxiliary memory	194
axiomatic semantics	222

B

backbone subsystem	202
Backus normal form, BNF	3
Backus-Naur formalism, BNF	3
back-propagation learning	159
bar code reader	771
BASIC language	1099
basic method of Chinese character recognition	275
basic operations in image processing	795
basic services of Internet	1113
batch processing	553
BCY language	1100
belief	908
benchmark	314
binary arithmetic operation	155
binding	9
biocomputing	653
biometrics	654
Boltzmann machine	33
Boolean algebra	37
Border Gateway Protocol, BGP	11
bottom up method	1068
bridge	854
bridge	854
broadband access architecture	486
broadband access network	487
broadcast routing	250
browser	509
browser - Web - database mode	509
bulletin board system, BBS	221
bus arbiter	1082
bus controller	1082
bus standard	1081
Business Process Reengineering, BPR	562

C

C language	1102
C ++ language	1100
cable access technologies	118
cabling system standard	39

cache coherence	208
cache	206
calculus of communication systems, CCS	778
camera calibration	648
campus subsystem	420
capability maturity model, CMM	545
cardinal number	306
cartoon animation	456
cartridge tape drive	278
case grammar	215
case-based reasoning, CBR	308
category theory	159
CD/DVD mirror server	243
cell switching	912
cell switching	912
cellular automaton	887
central processing unit, CPU	1051
centralized network management	334
cerebellar model articulation controller, CMAC	904
character set	1066
charge-coupled device memory	117
Chebyshev approximation	569
Chinese character calligraphic and phonological input method	276
Chinese character coding(keyboard) input method	269
Chinese character component input method	271
Chinese character input via keyboard	271
Chinese character phonological and calligraphic input method	276
Chinese character recognition	273
Chinese character stroke input method	268
Chinese characters recognition postprocessing	274
Chinese information processing	1050
Chinese information retrieval	1050
Chinese Ideograms Coded Characters Set for Information Interchange - Extension for the Basic Set, GB 18030,	911
Chinese Ideograms Coded Characters Set for Information Interchange-Basic Set, GB 2312—1980	910
Chinese pinyin input method	263
Chinese postman problem	1047

-
- Chinese speech recognition 265
- Chinese speech understanding 264
- Chomsky hierarchy 568
- Chomsky normal form 569
- CIP-L language 1101
- circuit switching 118
- class NP of problems 1121
- class P of problems 1132
- class 489
- classification of Chinese speech recognition 267
- classified criteria for security protection of
 computer information system 391
- cleaning disk 570
- client /server computing 478
- client/server mode 479
- cluster computing 331
- coaxial cable 782
- COBOL language 1101
- code generation 106
- code optimization 107
- coded Chinese character set, coded
 Ideographic character set 269
- coding of full band digital audio 572
- coding tool 12
- color management 937
- color model 938
- color reproduction 936
- combinatorial algorithm 1084
- combinatorics 1085
- combinatory logic 1084
- command language 528
- command, control, communication and
 intelligence system 1037
- Common Management Information Protocol, CMIP
 221
- commonsense 54
- communicating sequential processes, CSP 777
- communication control unit 776
- communication controller 776
- communication server 776
- compact disc-recordable, CD-R 739
- compact disc-rewritable, CD-RW 740
- compiler 12
- compiler-compiler 13
- complete partial order, CPO 826
- complex instruction set computer, CISC 198
- complexity measure 196
- complexity reduction 197
- component burn-in 995
- component reliability 995
- component screening 996
- component testing 994
- component-based software development method,
 CBSD 309
- composite theoretical performance, CTP 1079
- composite type 229
- compression and coding of digital voice 732
- compression and coding of images 796
- compression and coding standards for motion
 image 1003
- compression and coding standards of still images
 437
- computability theory 470
- computable function 469
- computational complexity theory 336
- computational geometry 409
- computational linguistics 414
- computational number theory 412
- computer aided design, CAD 352
- computer aided drafting 349
- computer aided manufacturing, CAM 353
- computer aided process planning, CAPP 348
- computer aided quality control, CAQ 354
- computer aided software engineering, CASE 352
- computer algebra 343
- computer animation 345
- computer assisted instruction, CAI 351
- computer availability 362
- computer category 365
- computer cluster 342
- computer control 363
- computer game 401
- Computer Graphics Interface Standard, CGI ... 1101
- Computer Graphics Metafile Standard, CGM 818
- computer graphics standard 378
- computer graphics 379
- computer hardware reliability 400
- computer hardware system testing 402

computer hardware	396
computer information system	390
computer integrated manufacturing system, CIMS	360
computer maintainability	361
computer maintenance	385
computer mathematics	376
computer music	394
computer network	380
computer organization	406
computer performance evaluation)	391
computer pipeline	367
computer reliability avai-lability and serviceability	339
computer room facility	359
computer security	341
computer simulation	346
computer software	369
computer supported cooperative work, CSCW	404
computer virus	342
computer vision	374
computerized process control manner	358
computerized process control	357
concentrator	334
concurrency control	18
concurrent engineering	26
concurrent programming language	18
concurrent programming	17
configuration management tool	552
congruence	782
connectionist expert system	500
constant	54
constraint reasoning	1002
content-based retrieval	311
context analysis	984
context free grammar	643
context sensitive grammar	644
continuous acquisition and life - cycle support, CALS	73
continuous system simulation	502
coprocessor	905
coroutine	906
corpus linguistics	988
corpus	986

courseware	479
cross compiler	423
cryptology	523
curve	571
customer engineer diskette	772

D

data circuit equipment, DCE	686
data communication interface standard	704
data communication interface	703
data communications	701
data dependency	707
data encoding	679
data flow diagram	699
data flow language	700
data integrity	706
data link control protocol	697
data link control	696
data mining	706
data model	700
data modulation	687
data path	701
data preparation device	707
data security	679
data sharing	688
data structured system development method	689
data structures	688
data structure-oriented method	526
data terminal equipment, DTE	707
data transfer	686
data transmission	682
data type	696
data warehouse	681
data	679
database administration	691
database connectivity standard	693
database design	693
database management system, DBMS	692
database performance evaluation	696
database system	694
database	690
dataflow computer	698
deadlock	758

debugging tool	772	distributed expert system, DES	184
decidable problem	475	distributed he-terogeneous computer system	183
decision support system, DSS	452	distributed interactive simulation, DIS	176
declaration	756	distributed multimedia system	179
declarative language	757	distributed network management	183
declarative representation	60	distributed operating system	177
deductive database	941	distributed processing system	178
deductive synthesis method	941	distributed programming	177
denotational semantics	1036	distributed shared memory	180
design language	647	distributed software system	181
design of algorithm	762	divider	76
design specification	646	documentation language	868
designing tool	646	domain engineering	508
development process	456	domain name sysytem, NDS	993
device testing	645	domain theory	513
digital camera	747	DOS operating system	1103
digital computer	736	duration calculus	656
digital data network, DDN	743	dynamic random access memory chip	134
digital differential analyzer	745		
digital geometry processing	735	E	
digital image processing	744	editing digital audio	749
digital integrated circuit	732	editing digital video	742
digital library	743	editor	12
digital magnetic recording	722	Eiffel language	1105
digital PC camera	741	electrically erasable programmable read only memory chip	117
digital signal processor, DSP	748	electromagnetic compatibility, EMC	115
digital signatures	741	electronic commerce, EC	124
digital subscriber line, DSL	750	electronic computer	121
digital versatile disc, DVD	730	electronic data interchange, EDI	128
digital - to - analog converter, DAC	711	electronic design automation, EDA	126
digitizer	731	electronic dictionary	121
direct memory access, DMA	1028	electronic government	131
direct representation	1027	electronic mail, E-mail	130
directed graph	983	electronic pre-press processing	129
direct current power supply	1029	embedded computer	567
discrete event system simulation	493	embedded operating systems	567
discrete mathematics	498	encapsulation	191
display device	889	encoding method of optical recording	241
distance education, distance learning	998	encoding methods of digital magnetic recording	725
distributed computing environment	180	engineering database	217
distributed control system, DCS	332	Enterprise Resource Planning, ERP	562
distributed database system	182		
distributed database	182		

entity relationship diagram	663	Fieldbus Control System, FCS	892
entity-relationship model	662	file manager	875
equation	553	file sharing	874
equipment subsystem	645	file transfer	872
erasable programmable read only memory chip	468	file	872
error analysis of numerical computation	717	Fingerprint Recognition	1042
error correcting codes of digital magnetic recording	729	finite automaton, FA	981
error detection and control of data transmission	684	finite element method	979
error detection codes of digital magnetic recording	728	firewall	161
Ethernet	953	first order logic	945
evolutionary model	940	flash erasable programmable read only memory chip	482
exception handling	958	flexible manufacturing system, FMS	598
expert system development environment	1060	floating-point number standard	192
expert system	1058	floppy disk adapter	603
explanation mechanism	431	floppy disk controller	600
expression	16	floppy disk drive testing	602
extensible language	474	floppy disk drive, FDD	601
extensible markup language, XML)	474	floppy disk storage	600
exterior routing protocol	822	floppy disk testing	599
external storage device interface	822	flow control	510
external storage subsystems	824	formal design specification	916
Extranet	826	formal functional specification	914
F		formal language theory	916
face recognition	591	formal method	912
facial expression animation	17	formal semantics	919
factoring	959	formal specification	914
fast Ethernet	484	FORTTRAN language	1107
fast Fourier transform	483	fountain model	552
fault diagnosis of computers	355	FP language	1107
fault recovery	233	fragmentation	186
fault-tolerant computing	594	frame relay switching	1015
feature extraction of speech recognition	935	frame representation	487
feature selection	770	frequency division multiplexing, FDM	557
features for Chinese character recognition	276	front-end processor	566
FGSPEC language	1106	full-text retrieval	575
Fiber distributed data interface, FDDI	246	function and procedure	263
Fiber optical cable	242	functional language	225
fiber - to - the - curb, FTTC	246	functional programming language	262
field	992	functional programming	262
		functional specification language	225
		functional specification	225
		fuzzy logic	530
		fuzzy reasoning	531

G

- GaAs integrated circuit, GaAs IC 649
 GaAs memory 648
 game tree search 34
 gateway 830
 GBK Chinese Internal Code Extension Specification
 272
 general purpose register 780
 genetic algorithm 949
 genetic learning 952
 geographic information system, GIS 113
 geometric modeling 335
 Gigabit Ethernet 565
 global positioning system, GPS 573
 Global System for Mobile communication, GSM
 573
 Gopher Services 1108
 Gouraud shading 1109
 grammar 868
 graph algorithm 787
 graph theory 786
 Graphical Kernel System Graphics Standard 1108
 graphics clipping 817
 graphics primitive generation 818
 graphics transformation 817
 great common divisor 1088
 green computer 513
 Greibach normal form 215
 grid computing 827
 group 577
 groupware 578
 GSPEC language 1109
 Gödel numbering 213
 Gödel's completeness theorem 214
 hard disk controller 970
 hard disk drive testing 972
 hard disk storage 968
 harddiskdrive, HDD 971
 hardware description language, HDL 973
 hardware synchronization mechanism 974
 hardware verification 976
 hard-wired control unit 976
 heterojunction devices of SiGe/Si 1009
 heuristic search 563
 hidden line/surface removal techniques 903
 hierarchical database 47
 high level language 205
 high performance computing 212
 high speed digital signal transmission 210
 high speed local area network 209
 higher-order logic 205
 holographic memory 575
 home page 1057
 Hopfield neural network model 1110
 horizontal subsystem 752
 Horn logic 293
 hub 334
 human body animation 592
 human-aided machine translation, HMT 593
 human-computer interaction system 589
 human-computer interaction technology,
 or HCI technology 588
 hybrid automaton 292
 hybrid computational models 291
 hybrid computer 290
 hybrid fiber coaxial cable, HFC 290
 hyper text markup language, HTML 60
 hypertext 59
 hypothesis testing 416

H

- halting problem 775
 hand gestures recognition 670
 Hanzi font, Hanzi typeface 277
 Hanzi order 278
 Hanzi, Chinese character, Han character,
 Chinese Hanzi 267
 hard disk adapter 973

I

- Ideographic repertoire 277
 illumination model 249
 image acquisition 801
 image analysis 800
 image boundary representation 788
 image degradation 811
 image edge detection 789

image enhancement	816	input/output device interface	674
image feature extraction	809	input/output interface	674
image geometric feature representation	802	input/output manager	673
image geometric operation	803	input/output technique	674
image model	806	instruction format	1037
image moment representation	804	instruction level parallel processing, ILPP	1038
image morphological operation	814	instruction register	1039
image motion-blurred restoration	815	instruction set	1041
image neighborhood operation	806	integrated circuit manufacturing	323
image parallel processing	793	integrated circuit test program	319
image pixel classification	813	integrated circuit test system	321
image point operation	798	integrated circuit, IC	315
image reconstruction	794	integrated digital loop carrier, IDLC	327
image region representation	808	integrated services digital network, ISDN	1079
image region segmentation	809	intelligent robot	1043
image representation	792	interactive language	424
image restoration by inverse filtering	799	interactive television	424
image restoration by wiener filtering	812	interconnection network, ICN	281
image restoration	800	interface definition language	427
image segmentation	800	interface tool	432
image skeleton representation	801	intergrated circuits in power supply	120
image thinning	813	interior routing protocol	544
image transform operation	792	Internet address	1112
image transformation	790	Internet architecture	1114
image understanding systems	805	internet protocol, IP	830
image-based modeling, IBM	313	Internet service provider, ISP	1112
image-based rendering, IBR	312	Internet	1111
impact printer	295	internetworking equipment	840
imperative language	528	internetworking protocol	840
implicit function surface	959	internetworking techniques	839
in-circuit testing of printed circuit board	965	internetworking	839
index	768	interpolation	48
inductive inference	252	interpreter	431
inductive learning	253	interrupt	1047
inductive synthesis method	255	Intranet	544
industrial control computer	219	intuitionistic logic	1028
inference engine	819	iris recognition	280
information hiding	912	iso-surfaces construction technique	111
information retrieval method	909		
information	909		
inheritance	414		
input device	672		
input devices of virtual reality	924		
input/output channel	677		

J

Jackson system development method	1115
Java language	1116
job	1093
job manager	1093

joystick 481

K

Kerberos authentication 1117
 key 422
 keyboard 423
 keyframe animation 233
 knowledge acquisition tool 1023
 knowledge acquisition 1021
 knowledge base 1026
 knowledge engineering 1020
 knowledge refinement 1025
 knowledge representation 1017

L

language model of speech recognition 936
 language processing system 989
 Larch language 1117
 large-scale computer, mainframe 104
 laser beam image typesetter 315
 lattice 214
 layout technology 38
 learning by analogy 490
 learning computation theory 930
 leased line network 1083
 legal protection of computer software 372
 lexical analysis 84
 light pen 239
 linear bounded automaton 898
 linear grammar 898
 linear logic 897
 linkage editor 500
 LINPACK benchmark 1118
 Linux operating system 1118
 LISP language 1119
 list processing language 15
 Listserv 1120
 literate programming 869
 loader 1064
 local area network reference model 439
 local network 437
 local search algorithm 437
 logic inference machine 517
 logic operation 517

logic programming language 516
 logic programming 515
 logic representation 514
 logic synthesis technologies for digital system ... 746
 logical link control sublayer of local area network
 440
 low earth orbit communication system 112
 low level language 113
 LR(k) Grammar 1120
 luminance calculation 243

M

machine cycle 305
 machine language 305
 machine learning 302
 machine translation evaluation 298
 machine translation, MT 296
 machine-aided human translation, MAHT 306
 macroprocessor 280
 magnetic card reader 87
 magnetic disk array 89
 magnetic disk storage 88
 magnetic storage 84
 magnetic tape storage 85
 magneto-optical disc drive 86
 main memory, MM 1057
 maintenance process 864
 maintenance tool 863
 Management Information Base, MIB 236
 management process 236
 many-sorted algebra 143
 mapping 968
 markup language 1046
 Martin-Löf's type theory 520
 massively parallel processing, MPP 101
 mathematical logic 708
 mathematical statistics 710
 matrix computation 445
 maximum flow 1088
 mean time between failures, MTBF 559
 mechanical theorem proving 132
 media processor 520
 medium access control method of local area network
 439

medium access control sublayer of local area network	440
memory error checking and correction	94
memory management unit, MMU	94
memory management	92
memory manager	93
memory organization	96
memory performance	96
memory protection	91
memory system	98
memory type	95
message handling system, MHS	901
message passing	902
metalanguage	996
method of least squares	1089
method of speech synthesis	934
metropolitan area network	61
micro packaging technology	862
micro-controller	858
micro-programmed control unit, MCU	856
microcomputer	860
microkernel	860
microprocessor	857
microwave communication	856
military command information system	454
minicomputer	905
minimum spanning tree	1090
Miranda language	1120
ML language	1121
Mobile Communication Network	949
mobile communication	949
mobile computer	947
mobile database	948
modal logic	539
model checking	541
model theory	542
modeling methodology of discrete event system simulation	495
modeling of natural phenomena	1073
modeling technique	1008
models of concurrency	19
modem	774
modular method	533
modular structure diagram	534

Modula-2 language	1121
Monte Carlo method	522
morphological analysis of natural language processing	1075
motion analysis	1003
mouse	677
multichip module	153
multimedia authoring tool	148
multimedia database	146
multimedia document	146
multimedia technology	144
multimodal human-computer interaction	148
multiple issue architecture	141
multiple valued logic	154
multiplexing	144
multiplier	62
multiprocessing system bus	139
multiprogramming	141
multi-modal fusion in biometrics	149
multi-resolution modeling	142

N

natural deduction method	1074
natural language processing, NLP	1074
natural language understanding, NLU	1077
network adapter	845
network application service	852
network architecture	847
network attached storage, NAS	194
network computing mode	841
network data unit	845
network database	846
network database	855
network design	843
network differential services, diffserv, DS	841
network engineering	836
network integrated services	841
network intrusion detection	842
Network Management Standard	837
network management	836
network middleware	853
network operating system, NOS	832
network operation environment	852

-
- | | | | |
|--|------|---|------|
| network planning | 838 | object | 138 |
| network processor | 833 | object-relation database | 138 |
| network protocol conformation testing | 851 | object-oriented method | 524 |
| network protocol engineering | 849 | object-oriented programming | 524 |
| network protocol formal description technology | 850 | Occam language | 1125 |
| network protocol specification | 850 | office automation | 3 |
| network protocol | 848 | off-line equipment | 820 |
| network resource reservation protocol, RSVP | 853 | off-line handwritten Chinese characters recognition | 820 |
| network security | 831 | off-line printer | 820 |
| network services | 834 | on-line analytical processing, OLAP | 503 |
| network software | 842 | online handwritten Chinese characters recognition | 505 |
| network test | 832 | online transaction processing, OLTP | 505 |
| neural computing | 651 | Open Shortest Path First Protocol | 1127 |
| neural network chip | 652 | open system interconnection reference model | 461 |
| new generation internet protocol, IPv6 | 907 | open system | 458 |
| non-monotonic logic | 167 | OpenGL Graphics Standard | 1126 |
| non-monotonic reasoning | 168 | operating system | 42 |
| nonimpact printer | 171 | operation process | 1006 |
| nonprocedural language | 171 | operational semantics | 45 |
| non-photorealistic rendering, NPR | 175 | operations of sets | 330 |
| non-traditional computer | 165 | optical character reader, OCR | 249 |
| normal form | 161 | optical computer | 240 |
| normalization | 256 | optical disc controller | 244 |
| notebook computer | 10 | optical disc library | 245 |
| NP complete problem | 1122 | optical disc tower | 245 |
| number system | 720 | optical mark reader, OMR | 248 |
| numerical approximation | 712 | optical recording | 241 |
| numerical computation | 715 | optical storage | 239 |
| numerical differentiation | 719 | optimization method | 1091 |
| numerical integration | 713 | optoelectronic integrated circuit, OEIC | 240 |
| numerical solution for system of linear algebraic equations | 894 | ordinal number | 929 |
| numerical solution for system of nonlinear algebraic equations | 174 | OSI management architecture | 1126 |
| numerical solution of matrix eigenvalue problems | 447 | output analysis of discrete event system simulation | 497 |
| numerical solution of ordinary differential equation | 55 | output device | 672 |
| numerical solution of partial differential OBJ language | 1125 | output devices of virtual reality | 923 |
-
- O**
- | | |
|--------------------------------|-----|
| object oriented database | 525 |
| object oriented language | 526 |
-
- P**
- | | |
|---|-----|
| packet assembler /disassembler, PAD | 190 |
| packet switching | 188 |
| page description language | 944 |
| page layout software | 547 |

parallel algorithm	27	power supply system for large-scale computer ...	105
parallel database	27	power supply testing	119
parallel processing system	22	pragmatics	992
parallel programming language	22	prepress image processing technology	960
parallel simulation	25	pre-and post-assertion method	566
parallelizing compiler	20	pre-editing	959
parameter estimation	40	primality test	759
parameterization for 3D mesh model	639	prime number	758
parametric curve	41	primitive recursive function	997
parametric surface	40	primitive type	419
PARLOG language	1128	printed circuit board design	963
PASCAL language	1129	printed circuit board testing	962
pattern classifier	536	printed Hanzi recognition	961
pattern recognition	537	printed page layout analysis	961
PDL language	1129	printer testing	101
perceptron	203	private branch exchange, PBX	1063
performance simulation of computer systems	388	private key cryptography	757
personal communication network, PCN	216	probabilistic algorithm	199
personal digital assistant, PDA	216	probabilistic automaton	199
Petri net theory	549	problem-oriented language	527
phase change disc drive	899	procedural implementation method	259
Phong shading	1131	procedural language	261
phrase structure grammar, PSG	137	procedural programming	260
phrase structure grammar	137	procedure representation	257
Physical medium - dependent sublayer of local area network, PMD	443	process algebra	433
physical medium-independent sublayer of local area network	442	process	432
physically based animation	313	processing data rate, PDR	682
physically-based modeling	314	processor architecture	78
planar graph	559	processor manager	80
plenoptic function based rendering	311	process-control computer	257
plotter	288	Product Data Exchange Specification, PDES	51
pointer type	1043	product data management, PDM	50
point-to-point protocol, PPP	115	production representation	52
Polish notation	31	production	52
polymorphic type	150	program analysis	64
polynomial hierarchy	151	program counter	64
polynomial space reduction	151	program logic	67
polynomial time reduction	152	program transformation method	72
Post machine	32	program	63
Post's correspondence problem	32	programmable controller, PC	466
post-editing	959	programmable read only memory chip	467
power supply for computer	344	Programmer's Hierarchical Interactive Graphics System, PHICS	1130
		programming language	69

programming methodology	68
programming	68
project management tool	900
PROLOG language	1131
propositional logic	529
Protocol Standards of Local Area Network—	
IEEE 802 Std	443
PSA program	1131
pseudo-random numbers	865
PSL language	1132
public data network, PDN	223
public key cryptography	224
public key infrastructure, PKI	224
public switched telephone network, PSTN	223
pushdown automaton	888

Q

qualitative reasoning	133
quality of network services	835
quantitative linguistics	336
quantum computing	507
query optimization	50
query processing	49
queueing theory	548

R

radiosity technique	193
random access machine, RAM	767
random access memory chip	766
range image acquisition and analysis	451
rapid prototyping method	997
ray tracing technique	247
read only memory chip	1032
read only memory, ROM	1032
read only optical disc drive	1035
real time rendering	662
realistic graphics generation	1011
record type	414
recursive function	114
reduced instruction set computer, RISC	433
reduction machine	255
reel-to-reel tape drive	463
regression analysis	285
regular expression	1012

regular grammar	1013
relation	233
relational algebra	234
relational database	235
relevance feedback in information retrieval	909
reliability design	473
reliability of computer system	387
Remote netwrk MONitoring, RMON	999
remote sensing information processing	943
rendezvous	287
repeater	1048
requirements analysis tool	928
requirements definition language	926
requirements engineering	929
resolution method	251
resource - sharing mode	1065
rewriting rule method	73
ring	285
robot control	300
robot motion planning	301
robot sensor	299
robotic vision	300
router	511
routine	499
Routing Information Protocol	1132
routing	512
ruggedization technology	416

S

satellite communication	866
satellite technologies	866
scanner	642
schema	536
security evaluation criteria of information system	
.....	912
security mechanisms	1
security services	1
self-checking circuit	1073
self-compiler	1067
self-organizing mapping model	1078
semantic analysis	991
semantic network representation	992
semantics	991
semiconductor memory chip	6

semiconductor memory	4	software methodology	606
semiconductor read-write memory	7	software middleware	633
semistructured data	9	software package	605
sequential control	754	software pipelining	625
sequential programming	754	software process model	618
serial transmission	83	software process	617
server	191	software quality	632
set theory	328	software reengineering	632
set	328	software reliability	623
SETL language	1133	software requirements definition	630
severe environment computer	465	software reuse	608
shared memory	228	software reverse engineering	626
shortest path problem	1089	software safety	603
sieve method	642	software systems	629
simple network management protocol,SNMP	419	software testing	605
simplification for 3D mesh model	640	software tool	614
simulation language	163	software understanding;software comprehension	624
simulation technologies for digital system	747	solderless crimp connection	878
simultaneous peripheral operations online, spool	418	solid state disk,SSD	230
single loop digital controller	109	solution of polynomial equation	204
single-chip computer	110	sorting algorithm	548
Smalltalk language	1133	space complexity	480
smart card reader	1045	spatial database	481
SNOBOL language	1134	speaker recognition	755
soft object animation	14	SPEC benchmark	1135
software agent	634	specifications of multimedia document	147
software architecture	628	speech synthesis of Chinese	264
software automation method	635	speech synthesizer	934
software component library	616	spiral model	518
software component	615	spline function	942
software configuration management	626	stack automaton	1009
software debugging	606	Standard generalized markup language,SGML	15
software design pattern	627	standard tape	14
software development environment	621	state transition diagram	1064
software development method	619	statement	985
software development model	622	static random access memory chip	435
software engineering economics	613	stereo vision	499
software engineering environment	612	storage area network,SAN	96
software engineering	609	store and forward switching	99
software language	631	streaming media	510
software library	624	streaming tape drive	511
software life cycle	627		
software maintenance	628		

string processing language	82	encission and spurious transmission, TEMPEST	162
structured analysis and design technique, SADT	430	technologies for computer applications	395
structured cabling system, SCS	428	technology for computer aided something	350
structured light method in computer vision	375	Telecommunication Management Network, TMN	119
structured method	429	telemedicine	1000
structured programming	429	temporal logic	660
subdivision surface	560	terminal device	1055
subprogram	1065	terminal server	1055
subroutine	1066	terminal	1055
Sun's theorem	768	testing tool	46
supercomputer	449	text to speech, TTS	875
superconducting integrated circuit	57	texture generation	876
supermode architecture	59	texture mapping	876
super-minicomputer	58	the standard for electronic commerce	124
supply process	227	theory of computation	410
supporting process	1016	theory of NP completeness	1123
surface mounting technology, SMT	16	theory of programs	65
surface	570	theory of vision computing	664
switch	425	thermal management, thermal design	580
switched multi-megabit data service, SMDS	426	thread	893
switching technologies	426	three dimensional shape recovering	640
synchronous optical network, SONET	781	three-level architecture of database systetime complexity	659
synchronous transmission	780	time division multiplexing, TDM	657
syntactic analysis of natural language processing	1076	timer	336
syntactic pattern recognition method	448	time-sharing mode	187
syntax	983	time-sharing processing	186
syntax analysis, parsing	983	timing system	661
syntax diagram	984	token bus network	577
system bus	886	token ring network	576
system compatibility	883	toolkit	218
system maintenance	884	top-down method	1068
system on a chip, SOC	553	topology of local area network	441
system performance specification	885	touch screen	80
systems programming language	883	TPC benchmark program	1137
		track ball	217
T		training simulator	932
tailoring process	418	transaction process monitor	663
task scheduler	593	transaction processing	663
TCP/IP protocol suite	1135	transaction	664
technique of electro-mechanical protection against			

UV

very long instruction word processor	1140
very small aperture terminal, VSAT	652
video compact disc player	665
video random access memory chip	667
videoconferencing	666
videophone	477
Vienna development method, VDM	864
view interpolation	668
virtual machine	921
virtual memory	919
virtual private network, VPN	926
virtual reality, VR	922
virtual terminal, VT	925
visual programming language	475
visual programming	476
visualization in scientific computing	465
visualization of data sets in 3D space	638
VLSI algorithm	1141
volume rendering technique	770

W

waterfall model	560
wavelength division multiplexing, WDM	30
wave-soldering	30
weakest precondition method	1089
wearable computing	469
web data management	827
wide area information server, WAIS	251
wide area network, WAN	250
wide spectrum language	250
wideband access technologies	486
window system	83
Windows operating system	1143
wireless application protocol, WAP	880
wireless local area network	878
wire-wrap connection	580
work area subsystem	219
workstation	220
world wide web, WWW	826
Wu method	877

X

X.75 Recommendation 1145

XCY language 1146
 XYZ /E language family 1146

Z

Z language 1146

λ

λ calculus 1148

π

π -calculus 1150

ω

ω -finite state automaton 1151

Σ

Σ (signature) algebra 1152

3

3D computer animation 638

3D mesh processing 639

内 容 索 引

说 明

1. 本索引是全书条目和释文中的主题词索引。索引主题按汉语拼音字母的顺序排列。第一字同音时,按其音调四声顺序排列;同音同调时依次按笔画多少和笔顺排列;如完全相同,则按第二字,余类推。非汉字开头的索引主题排在汉字主题后。依次为英文字母、希腊字母和阿拉伯数字开头的索引主题,它们分别按其字母顺序和数的顺序排列。

2. 设有条目的主题用黑体字印出,未设条目的主题用仿宋体字印出。

A

阿贝尔群·····	578	饱和度·····	939
阿克曼函数 ·····	1	保护键·····	91
安全服务 ·····	1	保证型服务·····	841
安全机制 ·····	1	报文交换·····	100
安全中间件·····	832	倍频码·····	726
按名调用·····	754	被标识对象·····	1043
按内容寻址存储器·····	506	被管对象·····	1127
按值调用·····	754	被管资源代理·····	837
按值-结果调用·····	754	本地终端·····	1055

B

巴克斯-诺尔形式体系 ·····	3	笔记本电脑 ·····	10
巴克斯-诺尔形式·····	3	闭式子例程·····	1066
巴克斯范式 ·····	3	边界网关协议 ·····	11
白盒测试·····	605	编辑程序 ·····	12
板级总线·····	886	编码·····	596
办公自动化 ·····	3	编码工具 ·····	12
半导体存储器 ·····	4	编码解码器·····	683
半导体存储器芯片 ·····	6	编译程序 ·····	12
半导体读写存储器 ·····	7	编译程序的编译程序 ·····	13
半导体盘·····	230	编译程序的生成程序·····	13
半结构化数据 ·····	9	编译时刻·····	990
半群·····	577	变尺度方法·····	1091
半色调加网·····	960	变换群·····	578
绑定 ·····	9	变量 ·····	14
		变量说明·····	756
		变体记录·····	414
		变形体动画 ·····	14
		变址寄存器·····	780

- | | | | |
|------------------|-----|----------------|------|
| 标准带 | 14 | 布思算法 | 157 |
| 标准幅度带 | 15 | 布图技术 | 38 |
| 标准扭斜带 | 15 | 布线系统标准 | 39 |
| 标准速度带 | 15 | | |
| 标准通用置标语言 | 15 | C | |
| 表处理语言 | 15 | 裁剪 | 804 |
| 表达式 | 16 | 彩色图像 | 807 |
| 表面安装技术 | 16 | 参数传递 | 263 |
| 表情动画 | 17 | 参数估计 | 40 |
| 并发程序 | 433 | 参数曲面 | 40 |
| 并发程序设计 | 17 | 参数曲线 | 41 |
| 并发程序设计语言 | 18 | 参照完整性 | 706 |
| 并发控制 | 18 | 残片 | 92 |
| 并发模型 | 19 | 操作存储器 | 1057 |
| 并行编译程序 | 20 | 操作码 | 1037 |
| 并行程序设计 | 21 | 操作系统 | 42 |
| 并行程序设计语言 | 22 | 操作型处理 | 681 |
| 并行处理系统 | 22 | 操作语义 | 45 |
| 并行处理系统加速比 | 392 | 操作指导 | 358 |
| 并行仿真 | 25 | 操作指导方式 | 363 |
| 并行工程 | 26 | 测量法 | 391 |
| 并行排序 | 549 | 测试工具 | 46 |
| 并行排序算法 | 549 | 测试计划 | 46 |
| 并行数据库 | 27 | 测试计划辅助工具 | 47 |
| 并行算法 | 27 | 测试夹 | 966 |
| 波分多路复用 | 30 | 测试结果分析程序 | 47 |
| 波峰焊 | 30 | 测试码生成 | 356 |
| 波兰记法 | 31 | 测试数据生成程序 | 47 |
| 波斯特代数 | 154 | 测试支持环境 | 47 |
| 波斯特对应问题 | 32 | 层次结构存储器 | 98 |
| 波斯特机 | 32 | 层次模型 | 47 |
| 玻耳兹曼机 | 33 | 层次数据库 | 47 |
| 博弈树搜索 | 34 | 插入排序 | 549 |
| 补码 | 721 | 插入损耗 | 30 |
| 补元 | 214 | 插值 | 48 |
| 不对称数字用户专用线 | 751 | 插值方法 | 803 |
| 不归零制 | 725 | 插值三次样条函数 | 942 |
| 不恢复余数法 | 76 | 查全率 | 909 |
| 不间断电源 | 35 | 查询处理 | 49 |
| 不可判定问题 | 37 | 查询优化 | 50 |
| 不认人语音识别 | 267 | 查准率 | 909 |
| 布尔代数 | 37 | 差分法 | 554 |
| 布尔型 | 419 | 产品数据管理 | 50 |
| 布罗依登法 | 175 | 产品数据交换标准 | 51 |

产生式	52	程序验证	70
产生式表示	52	程序再定位	93
常量	53	程序转换方法	72
常识	54	持续采办和全寿命支持	73
常识推理	54	抽象代数	74
常微分方程数值解法	55	抽象数据类型	75
场效应晶体管	315	出度	983
超标量处理机	1140	除法器	76
超标量结构	142	储存模式	691
超长指令字	22	处理机体系结构	78
超长指令字结构	142	处理器管理程序	80
超大规模集成电路	732	触发器	733
超导集成电路	57	触觉与力觉反馈装置	924
超级小型计算机	58	触屏	80
超结点结构	59	穿通	440
超类	526	传输控制协议	80
超立方体网	282	传输时间	89
超流水线结构	142	传输损耗	81
超链接	60	传统密码	523
超流水线处理机	1141	传统密钥	523
超媒体	59	串处理语言	82
超穷基数	306	串归约机	256
超文本	59	串扰衰减	30
超文本传送协议	827	串行传输	83
超文本置标语言	60	串行访问存储器	95
超越扩张	993	串行排序算法	549
陈述性表示	60	窗口	83
成组编码	727	窗口系统	83
成组多路通道	677	垂直磁记录	724
城域网	61	垂直分片	186
乘法器	62	纯度	939
乘幂法	447	词标	84
程控交换机	223	词法分析	84
程控模拟交换机	223	词法分析程序的生成程序	14
程控数字交换机	223	磁表面存储器	85
程序	63	磁存储器	84
程序分析	64	磁带存储器	85
程序计数器	64	磁带机	85
程序理论	65	磁带排序	549
程序逻辑	67	磁带驱动器	85
程序切片分析工具	498	磁道	88
程序设计	68	磁电阻式随机存取存储器(MRAM)芯片	1035
程序设计方法学	68	磁放大器电源	1029
程序设计语言	69	磁杆存储器	84

磁鼓存储器	84	代数闭域	993
磁光盘驱动器	86	代数规约	107
磁卡	87	代数结构	74
磁卡机	87	代数扩张	993
磁膜存储器	84	代数语义	108
磁盘存储器	88	带通滤波器	557
磁盘排序	549	带优先次序的调度问题	1122
磁盘阵列	89	戴德金格	215
磁头	85	单步法	56
磁心存储器	84	单发射结构	141
存储保护	91	单回路数字控制器	109
存储单元	8	单极型晶体管	315
存储管理	92	单扩张	993
存储管理程序	93	单片计算机	110
存储间距	96	单片微型计算机	858
存储键	91	单元测试	46
存储结构	688	单元刚度矩阵	980
存储密度	88	道密度	88
存储面数	88	等待时间	89
存储器差错校验	94	等速度可扩展性度量	29
存储器管理部件	94	等效率可扩展性度量	29
存储器类型	95	等值面构造技术	111
存储器性能	96	低轨道卫星通信系统	112
存储器总线	139	低级语言	113
存储器组成	96	低速局域网	438
存储区域网	96	低通滤波器	557
存储容量	8,88	笛卡儿积	331
存储系统	98	底图	786
存储周期	8	地理信息系统	113
存储转发	440	地址寄存器	662
存储转发交换	99	地址解析	1113
存取时间	96	地址解析协议	1113
D		地址码	1038
打印服务器	192	递归	114
打印机测试	101	递归函数	114
大规模并行处理	101	递归类型	230
大规模集成电路	732	第四代风范	611
大型计算机	104	点对点连接协议	115
大型计算机电源系统	105	点阵并行击打印字设备	296
代理	1127	点阵串行击打印字设备	295
代理服务器	161	点阵击打式印字设备	295
代码生成	106	电磁干扰	116
代码优化	107	电磁兼容性	115
		电荷耦合器件存储器	117

- 电可擦编程只读存储器芯片 117
- 电可修改只读存储器芯片 1033
- 电缆接入技术 118
- 电路交换 118
- 电信管理网络 119
- 电源测试 119
- 电源集成电路 120
- 电子词典 121
- 电子计算机 121
- 电子记事本 216
- 电子商务 124
- 电子商务标准 124
- 电子设计自动化 126
- 电子数据交换 128
- 电子印前处理 129
- 电子邮件 130
- 电子杂志 1120
- 电子政务 131
- 调试程序生成程序 772
- 调试工具 772
- 调整盘 772
- 调制 774
- 调制解调器 774
- 迭代法 174
- 迭代风范 611
- 叠堆式集线器 334
- 定点表示法 737
- 定点数 721
- 定理机器证明 131
- 定时器 336
- 定性视觉、主动视觉 375
- 定性推理 133
- 定义域 968
- 动画后期处理 133
- 动画描述语言 134
- 动态绑定 9
- 动态测试程序 46
- 动态程序再定位 93
- 动态路由算法 512
- 动态数组 751
- 动态随机存取存储器芯片 134
- 动态装入程序 1064
- 度 882
- 短语结构文法 137
- 短语结构语法 137
- 段保护 91
- 段表 920
- 段表项 920
- 段式存储系统 99
- 段页式存储系统 99
- 断言 914
- 断言处理程序 46
- 堆栈指针 1040
- 对比度 807
- 对称密钥密码系统 523
- 对称群 578
- 对称式多处理机 24
- 对分宽度 282
- 对偶范畴 160
- 对偶原理 214
- 对象 - 关系数据库 138
- 对象 138
- 多步法 56
- 多层前向网络 584
- 多处理机系统总线 139
- 多带图灵机 337
- 多道程序设计 141
- 多端口视频随机存取存储器芯片 667
- 多发射结构 141
- 多分辨率造型 142
- 多功能卡 603
- 多级互联网 283
- 多级缓存 96
- 多类代数 143
- 多路复用 144
- 多路转换器通道 677
- 多媒体技术 144
- 多媒体数据库 146
- 多媒体文档 146
- 多媒体文档规范 147
- 多媒体著作工具 148
- 多模态人机交互 148
- 多模态生物特征融合 149
- 多态类型 150
- 多线程 RISC 435
- 多线程处理机 80
- 多相整流电源 105
- 多项式环 285

- 多项式空间归约 151
 多项式谱系 151
 多项式时间归约 152
 多芯片模块 153
 多值逻辑 154
 多值依赖 707
 多重网格法 556
- E**
- 二进制算术运算 155
 二元关系 233
- F**
- 发光二极管印刷机 172
 翻译程序 989
 反变函子 160
 反传学习 159
 反汇编及高级程序设计语言的反编译工具 498
 反码 721
 反幂法 447
 反驱动 965
 反绎推理 253
 泛代数 74
 范畴 74
 范畴论 159
 范式 161
 防火墙 161
 防信息泄漏技术 162
 仿真语言 163
 访问监控器 391
 访问控制 164
 访问控制表 164
 访问域 443
 非 517
 非传统计算机 165
 非单调逻辑 167
 非单调推理 168
 非对称密钥密码系统 523
 非功能需求 926
 非过程语言 171
 非击打式印刷机 171
 非监督学习 173
 非结合环 285
 非均匀量化 807
- 非确定型图灵机 1122
 非特权状态 91
 非线性代数方程组数值解法 174
 非易失性存储器 1032
 非易失性存储器芯片 6
 非真实感图形绘制 175
 分辨率 808
 分波器 30
 分布交互式仿真 176
 分布式操作系统 177
 分布式程序设计 177
 分布式程序设计语言 182
 分布式处理系统 178
 分布式多媒体系统 179
 分布式共享存储 180
 分布式计算环境 180
 分布式软件系统 181
 分布式数据库 182
 分布式数据库系统 182
 分布式网络管理 183
 分布式文件系统 182
 分布式异构型计算机系统 183
 分布式专家系统 184
 分配格 214
 分配排序 549
 分片 186
 分散控制 358
 分色 937, 960
 分时操作系统 186
 分时处理 186
 分时共享模式 187
 分析型处理 681
 分析学习 187
 分支 882
 分子器件 653
 分组交换 188
 分组密码 757
 分组装拆器 190
 封装 191
 封装 422
 冯·诺依曼计算机 738
 逢1变化不归零制 726
 服务器 191
 服务原语 462

浮点表示法	737
浮点数	721
浮点数标准	192
浮动目标程序	500
浮动装入程序	1064
幅度调制	687
幅移键控	774
幅移键控法	687
辐射度技术	193
辅助存储器	194
辅助索引	768
附网存储	194
附属处理机	906
复合语句	985
复用器	30
复杂性	196
复杂性度量	196
复杂性归约	197
复杂指令集计算机	198
赋值语句	985

G

伽罗瓦域	993
伽玛校正	937
改进调频制码	726
概率分析法	719
概率神经网络	585
概率算法	199
概率自动机	199
概念模型	700
概念设计	694
概要设计工具	646
干线子系统	202
甘岑系统	897
感知机	203
高比特率数字用户专用线	750
高彩色	891
高次代数方程解法	204
高级汇编程序	286
高级语言	205
高阶逻辑	205
高可用计算机	367
高斯	894
高速缓冲存储器	206

高速缓冲存储器一致性	208
高速缓存	206
高速局域网	209
高速数字信号传输	210
高性能计算	212
哥德尔配数	213
哥德尔完全性定理	214
割线法	175
格	214
格雷贝奇范式	215
格雷码	720
格式化容量	88
格语法	215
隔位扫描技术	657
隔字符扫描技术	657
个人计算机	860
个人数字助理	216
个人通信网	216
个人通信业务	217
跟踪球	217
工程数据库	217
工具集成	614
工具箱	218
工业控制计算机	219
工作区子系统	219
工作站	220
工作站机群	342
工作站网络	342
公告板系统	221
公共管理信息协议	221
公共应用服务要素	852
公理方法	708
公理语义	222
公用交换电话网	223
公用数据网	223
公钥	523
公钥基础设施	224
公钥密码技术	224
公钥密码系统	523
功能测试	46
功能抽象	524
功能规约	225
功能规约语言	225
功能需求	926

功能语言·····	225	光线跟踪技术·····	247
供应过程·····	227	光学标记阅读机·····	248
共轭梯度法·····	1091	光学字符阅读机·····	249
共享存储·····	228	光栅·····	1101
共享存储并行程序设计·····	29	光照模型·····	249
构造类型·····	229	广播·····	250
构造语句·····	985	广播路由选择·····	250
孤立词识别·····	267	广播式通信·····	778
固定时间的加速比·····	29	广谱语言·····	250
固件·····	677	广义模糊逻辑·····	530
固态硬盘·····	230	广域网·····	250
故障定位·····	355	广域信息服务系统·····	251
故障恢复·····	233	归并排序·····	549
故障检测·····	355	归结方法·····	251
故障模拟·····	356	归结式·····	251
挂网·····	937	归结原理·····	132
关键帧动画·····	233	归纳推理·····	252
关系·····	233	归纳学习·····	253
关系代数·····	234	归纳综合方法·····	255
关系模型·····	235	归约机·····	255
关系数据库·····	235	规范化·····	256
管理程序·····	629	规格化·····	157
管理过程·····	236	规约对象·····	226
管理信息库·····	236	规约方法·····	226
管理信息系统·····	237	规约性质·····	227
管理子系统·····	238	轨迹法·····	255
管态·····	91	过程表示·····	257
光笔·····	239	过程分析·····	20
光存储器·····	239	过程改进·····	619
光电集成电路·····	240	过程集成·····	622
光计算机·····	240	过程建模·····	618
光记录·····	241	过程控制计算机·····	257
光记录编码方法·····	241	过程实现方法·····	259
光缆·····	242	过程式程序设计·····	260
光亮度计算·····	243	过程语句·····	985
光流法·····	1003	过程语言·····	261
光盘镜像服务器·····	243	过滤路由器·····	161
光盘控制器·····	244	过驱动·····	965
光盘库·····	245		
光盘塔·····	245		
光盘自动换盘机·····	245		
光纤到路边·····	246		
光纤分布式数据接口·····	445		
光纤分布数据接口·····	246		

H

函数副作用·····	263
函数式程序设计·····	262
函数式程序设计语言·····	262
函数依赖·····	707

函数与过程	263	环	285
汉语拼音输入法	263	环境保护	91
汉语言语(语音)合成	264	环同构	285
汉语言语(语音)理解	264	环同态	285
汉语言语(语音)识别	265	恢复余数法	76
汉语言语(语音)识别分类	267	回归测试	46
汉字	267	回归分析	285
汉字笔画输入法	268	回路	786
汉字编码(键盘)输入方法	269	回溯法	1002
汉字编码字符集	269	汇编程序	286
汉字部件(字根)输入法	271	汇编语言	287
汉字键盘输入	271	会合	287
汉字内码扩展规范—GBK	272	会话密钥	421
汉字识别	273	绘图机	288
汉字识别后处理	274	绘制	380
汉字识别基本方法	275	混成系统	913
汉字识别特征	275	混合光纤同轴电缆	290
汉字形音输入法	276	混合计算机	290
汉字音形输入法	276	混合计算模型	291
汉字字汇	277	混合型表达式	16
汉字字型	277	混合自动机	292
汉字字序	278	活动扫描法	496
豪斯霍尔德方法	448	活动图	430
合波器	30	活锁	975
合成映射	968	或	517
核心识别法	625	获取过程	292
盒带驱动器	278	霍恩逻辑	293
黑白图像(灰度图像)	807		
黑盒测试	605	J	
横向扫描记录	278	击打式打印机	295
横向转换	72	机密性	831
宏处理程序	280	机器翻译	296
宏调用	280	机器翻译评价	298
宏定义	280	机器人传感器	299
宏汇编程序	286	机器人控制	300
宏扩展	280	机器人视觉	300
虹膜识别	280	机器人运动规划	301
后继序号	929	机器视觉	374
呼叫虚电路	189	机器学习	302
互操作性	184	机器语言	305
互联网	281	机器周期	305
互联网络的企业	544	机助人译	306
滑动窗口流控	510	基-r FFT算法	484
划分问题	1122	基本输入输出系统	1103

基本数据模型	700	计算机安全	341
基地址寄存器	780	计算机病毒	342
基数	306	计算机簇	342
基于 Agent 的计算	307	计算机代数	343
基于案例的推理	308	计算机电源	344
基于构件的软件开发方法	309	计算机动画	345
基于合一的语法理论	310	计算机对象	138
基于类型理论的方法	310	计算机仿真	346
基于内容的检索	311	计算机辅助工艺规划	348
基于全视函数的绘制	311	计算机辅助绘图	349
基于特征的方法	1003	计算机辅助技术	349
基于图像的绘制	312	计算机辅助教学	351
基于图像的造型	313	计算机辅助软件工程	352
基于物理的动画	313	计算机辅助设计	352
基于物理的造型	314	计算机辅助制造	353
基准程序	314	计算机辅助质量控制	354
基准手册	868	计算机故障诊断	355
激光存储器	239	计算机过程控制	357
激光印刷机	172	计算机过程控制方式	358
激光照排机	315	计算机机房设施	359
吉布森混合法	392	计算机集成制造系统	360
极限序数	929	计算机可维护性	361
集成电路	315	计算机可用性	362
集成电路测试程序	319	计算机控制	363
集成电路测试系统	321	计算机类型	365
集成电路卡	1046	计算机流水线	367
集成电路制造	323	计算机软件	369
集成度	318	计算机软件的法律保护	372
集成数字环载波	327	计算机视觉	374
集合	328	计算机视觉中的结构光法	375
集合范畴	160	计算机数学	376
集合论	328	计算机数字控制	353
集合运算	330	计算机图形标准	378
集群式计算	331	计算机图形学	379
集散控制系统	332	计算机网络	380
集线器	334	计算机维护	385
集中器	334	计算机系统可靠性	387
集中式网络管理	334	计算机系统性能模拟	388
几何造型	335	计算机协同工作	404
计量语言学	336	计算机信息系统	390
计时器	336	计算机信息系统安全保护等级划分准则	391
计数器	735	计算机性能评价	391
计算复杂性理论	336	计算机音乐	394
计算机 RAS 技术	339	计算机应用技术	395

- 计算机硬件····· 396
- 计算机硬件可靠性····· 400
- 计算机游戏····· 401
- 计算机整机检测····· 402
- 计算机支持的协同工作····· 404
- 计算机组成····· 406
- 计算几何····· 409
- 计算理论····· 410
- 计算数论····· 412
- 计算语言学····· 414
- 记录类型····· 414
- 继承····· 414
- 继承性····· 524
- 寄存器····· 735
- 加法器····· 415
- 加固计算机····· 465
- 加固技术····· 416
- 加权图····· 786
- 加速比····· 29
- 假共享····· 209
- 假设检验····· 416
- 假脱机····· 418
- 监督控制····· 358
- 监督学习····· 304
- 监控帧····· 441
- 监视器····· 889
- 监听····· 208
- 剪裁过程····· 418
- 简单类型····· 419
- 简单图····· 786
- 简单网络管理协议····· 419
- 建筑群子系统····· 420
- 鉴别····· 421
- 键保护····· 91
- 键码····· 422
- 键盘····· 423
- 键盘-软磁盘录入设备····· 707
- 交叉编译程序····· 423
- 交叉存储器····· 96
- 交叉开关····· 284
- 交互式电视····· 424
- 交互式语言····· 424
- 交换方式····· 440
- 交换环····· 285
- 交换机····· 425
- 交换技术····· 426
- 交换排序····· 549
- 交换式多兆位数据业务····· 426
- 交换式集线器····· 334
- 交换虚电路····· 189
- 阶····· 786
- 阶码····· 721
- 接口····· 533
- 接口定义语言····· 427
- 接入网····· 487
- 结构····· 142
- 结构测试····· 46
- 结构化布线系统····· 428
- 结构化程序设计····· 429
- 结构化方法····· 429
- 结构化分析····· 620
- 结构化分析与设计技术····· 430
- 结构性规则····· 897
- 截获-播放工具····· 46
- 解调····· 774
- 解释程序····· 431
- 解释机制····· 431
- 界面工具····· 432
- 界限寄存器保护····· 91
- 金属-氧化物-半导体场效应晶体管····· 315
- 金属-氧化物-半导体存储器····· 5
- 紧密耦合并行处理系统····· 24
- 尽力而为服务····· 841
- 进程····· 432
- 进程代数····· 433
- 进程交互法····· 496
- 经典逻辑····· 897
- 晶体管-晶体管逻辑····· 732
- 精简指令集计算机····· 433
- 精密照排系统····· 130
- 景物····· 806
- 静电印刷机····· 172
- 静态绑定····· 9
- 静态程序再定位····· 93
- 静态分析程序····· 46
- 静态路由算法····· 512
- 静态数组····· 751
- 静态随机存取存储器芯片····· 435

静止图像	807
静止图像压缩编码标准	437
镜像变换	803
局部搜索算法	437
局部性原则	206
局部优化	107
局域网	437
局域网基准(参考)模型	439
局域网介质访问控制方法	439
局域网介质访问控制子层	440
局域网逻辑链路控制子层	440
局域网拓扑结构	441
局域网物理介质无关子层	442
局域网物理介质相关子层	443
局域网协议标准(IEEE 802 标准)	443
矩阵计算	445
矩阵特征值问题数值解法	447
巨大规模集成电路	733
巨型计算机	449
句法模式识别方法	448
句法树	1076
具体范畴	160
距离图像获取与分析	451
聚集索引	768
聚类分析	173
决策支持系统	452
绝对误差	718
绝对装入程序	1064
军事指挥信息系统	454
均方逼近	712
均匀量化	807

K

卡通动画	456
开发过程	456
开发信息库工具	863
开放数据库互连	693
开放体系结构	458
开放系统	458
开放系统互连	843
开放系统互连基准(参考)模型	461
开放应用系统	458
开关电源	1029
开盘式磁带驱动器	463

开式子例程	1066
抗恶劣环境计算机	465
科学归纳推理	252
科学计算可视化	465
可编程的片上系统	399
可编程控制器	466
可编程时序逻辑器件	735
可编程只读存储器芯片	467
可擦编程只读存储器芯片	468
可测性设计	356
可穿戴计算	469
可穿戴计算机	469
可达到性分析工具	646
可计算函数	469
可计算性理论	470
可靠性模型	473
可靠性设计	473
可扩充语言	474
可扩展置标语言	474
可满足性问题	1122
可判定问题	475
可伸缩一致性接口	284
可视编程语言	475
可视程序设计	476
可视电话	477
可信计算机系统评估准则	477
可信计算基	477
可用性	831
刻面分类方法	616
客户-服务器计算	478
客户-服务器模式	479
课件	479
空分交换机	118
空集	328
空间复杂性	480
空间灰度关系矩阵	811
空间数据	481
空间数据库	481
空语句	985
孔斯曲面	409
控制存储器	856
控制负载型服务	841
控制杆	481
控制集成	622

控制驱动	165
快擦写存储器	10
快可擦编程只读存储器芯片	482
快速傅里叶变换	483
快速矩阵乘法	445
快速算法	483
快速以太网	484
宽带接入技术	486
宽带接入体系结构	486
宽带接入网	487
宽度优先搜索	788
框架表示	487
扩域	993
扩展编译	259

L

蓝牙技术	879
类	489
类比推理	489
类比学习	490
类型定义	492
类型论	492
类型说明	492
累加器	780
离散控制系统	364
离散牛顿法	175
离散事件系统	493
离散事件系统仿真	493
离散事件系统仿真建模方法学	495
离散事件系统仿真输出分析	497
离散数学	498
离散相似法仿真	503
离线攻击	421
离线图灵机	337
理解工具	498
立方体连结环	282
立体视觉	499
立体显示装置	923
利用率	393
例程	499
粒度	681
连接编辑程序	500
连接程序	500
连接词识别	267

连接专家系统	500
连接装入程序	1064
连通图	786
连续统	306
连续系统仿真	502
连续语音识别	267
联邦型分布式数据库系统	183
联合评审过程	1017
联机分析处理	503
联机事务处理	505
联机手写汉字识别	505
联机系统	250
联想存储器	506
联想记忆	506
链接法	47
链路控制协议	115
链路状态路由算法	512
良序集	234
亮度	940
量子并行	507
量子计算	507
量子计算机	507
量子寄存器	507
量子逻辑门	507
量子器件	507
量子位	507
邻接法	47
领域工程	508
浏览器	509
浏览器-万维网-数据库模式	509
流控	510
流媒体	510
流式磁带驱动器	511
龙格-库塔公式	56
路径	786
路由表	512
路由器	511
路由选择	512
路由选择算法	512
旅行商问题	1122
绿色计算机	513
论域理论	513
逻辑表示	514
逻辑程序设计	515

- | | | | |
|----------------|-----|---------------|------|
| 逻辑程序设计语言····· | 516 | 命题逻辑····· | 529 |
| 逻辑地址····· | 92 | 模代数····· | 154 |
| 逻辑结构····· | 688 | 模调度法····· | 625 |
| 逻辑联结词····· | 897 | 模格····· | 215 |
| 逻辑模式····· | 691 | 模糊逻辑····· | 530 |
| 逻辑设计····· | 694 | 模糊推理····· | 531 |
| 逻辑推理机····· | 517 | 模块····· | 533 |
| 逻辑运算····· | 517 | 模块化方法····· | 533 |
| 螺旋模型····· | 518 | 模块汇编程序····· | 286 |
| 螺旋扫描记录····· | 278 | 模块结构图····· | 534 |
| M | | | |
| 马丁洛夫类型理论····· | 520 | 模块说明····· | 756 |
| 码分多址····· | 290 | 模拟计算机····· | 535 |
| 码组····· | 443 | 模拟式调整软磁盘····· | 772 |
| 脉冲代码调制····· | 680 | 模拟数据····· | 682 |
| 曼彻斯特编码····· | 680 | 模拟信号····· | 683 |
| 漫游····· | 879 | 模式····· | 536 |
| 猫眼盘····· | 772 | 模式分类器····· | 536 |
| 枚举归纳推理····· | 252 | 模式识别····· | 537 |
| 媒体处理器····· | 520 | 模数转换器····· | 539 |
| 门电路····· | 733 | 模态逻辑····· | 539 |
| 蒙特卡罗法····· | 522 | 模型法····· | 391 |
| 密码学····· | 523 | 模型检验····· | 541 |
| 密文····· | 523 | 模型论····· | 542 |
| 密钥分发协议····· | 421 | 摩尔定律····· | 316 |
| 密钥分发中心····· | 422 | 默认推理····· | 168 |
| 幂集····· | 331 | 目标程序····· | 13 |
| 面密度····· | 88 | 目标机····· | 423 |
| 面向对象····· | 524 | 目态····· | 91 |
| 面向对象程序设计····· | 524 | N | |
| 面向对象方法····· | 524 | 内部路由协议····· | 544 |
| 面向对象分析····· | 525 | 内存存储器····· | 1057 |
| 面向对象设计····· | 525 | 内涵数据库····· | 941 |
| 面向对象实现····· | 525 | 内联网····· | 544 |
| 面向对象数据库····· | 525 | 内码····· | 910 |
| 面向对象数据库管理····· | 526 | 内排序····· | 548 |
| 面向对象语言····· | 526 | 内总线····· | 886 |
| 面向数据结构方法····· | 526 | 能力成熟度模型····· | 545 |
| 面向问题语言····· | 527 | 能行性理论····· | 470 |
| 明度····· | 939 | 拟牛顿法····· | 174 |
| 明文····· | 523 | 逆向工程工具····· | 863 |
| 命令式语言····· | 528 | 牛顿法····· | 174 |
| 命令语言····· | 528 | 牛顿下山法····· | 174 |

O

偶检验 684

P

排版软件 547
排错程序 991
排队论 548
排序算法 548
胖树 282
抛弃策略 997
佩特里网论 549
配置管理工具 552
配置管理过程 1016
喷泉模型 552
碰撞域 485
碰撞域直径 485
批处理 553
批处理操作系统 553
片上高速缓存 96
片上系统(SoC)体系结构 469
片上系统 553
片总线 886
偏微分方程 554
偏微分方程数值解法 553
偏序集 234
拼贴 804
频带 683
频分多路复用 557
频分多址 290
频宽 683
频率抽取 484
频率调制 687
频谱 683
频移键控 774
频移键控法 687
平方逼近 558
平均故障间隔时间 559
平均情况复杂性 196
平均数据传送率 686
平均无差错时间 559
平均无故障时间 362
平均修复时间 362
平面图 559

平台 183
剖分曲面 560
瀑布模型 560

Q

奇检验 684
奇偶检验 684
企业流程重组 562
企业数据集成 129
企业资源规划 562
启发式搜索 563
千兆位以太网 565
签名 422
前端处理器 566
前后断言方法 566
嵌入式操作系统 567
嵌入式电子设备 567
嵌入式计算机 567
嵌入式控制器 970
强驱动 965
乔姆斯基层次 568
乔姆斯基范式 569
桥接局域网 438
切比雪夫逼近 569
清洗盘 570
情况语句 985
请求调页 93
区间分析法 719
区域分裂法 556
曲面 570
曲线 571
取数时间 8
取指周期 662
权标 440
权标传递 440
权标传递方法 576
权标环网 576
权标总线网 577
全加器 415
全局型分布式数据库系统 183
全局优化 107
全军用化计算机 465
全频带数字音频的编码 572
全球定位系统 573

全球信息基础设施	1112	软件安全性	603
全球移动通信系统	573	软件包	605
全文检索	575	软件测试	605
全文自动标引	1069	软件调试	606
全息照相存储器	575	软件度量	46
确定型图灵机	1122	软件方法学	606
确认过程	1017	软件风险	518
群	577	软件复用	608
群件	578	软件工程	609
群控	354	软件工程环境	612
群同构	578	软件工程经济学	613
群同态	578	软件工具	614
群同态定理	578	软件构件	615
R		软件构件库	616
绕接	580	软件规约	635
热电致冷器	581	软件过程	617
热管	582	软件过程成熟度	545
热设计	580	软件过程模型	618
人工神经网络	582	软件过程能力	545
人工神经网络在模式识别中的应用	584	软件开发方法	619
人工生命	586	软件开发环境	621
人工智能	586	软件开发模型	622
人机交互技术	588	软件可靠性	623
人机交互系统	589	软件库	624
人脸识别	591	软件理解	624
人体动画	592	软件流水	625
人助机译	593	软件逆向工程	626
认人语音识别	267	软件配置管理	626
任务调度程序	593	软件设计模式	627
冗余	596	软件生存周期	627
容错	594	软件体系结构	628
容错计算	594	软件退役	864
容错计算机	595	软件维护	628
柔性制造系统	598	软件系统	629
入度	983	软件需求定义	630
软磁盘	600	软件语言	631
软磁盘测试	599	软件再工程	632
软磁盘存储器	600	软件质量	632
软磁盘控制器	600	软件质量度量	633
软磁盘驱动器	601	软件中间件	633
软磁盘驱动器测试	602	软件主体	634
软磁盘适配器	603	软件自动化方法	635
软磁盘阵列	91	软中断	1040
		弱连通	983

S

- 三角分解法····· 894
- 三阶段法····· 496
- 三模冗余····· 596
- 三维动画····· 638
- 三维空间数据场可视化····· 638
- 三维网格处理····· 639
- 三维网格模型参数化····· 639
- 三维网格模型简化····· 640
- 三维形态恢复····· 640
- 扫描仪····· 642
- 色调····· 939
- 色度····· 940
- 色域····· 936
- 筛法····· 642
- 闪存芯片····· 6
- 扇区····· 88
- 商泛代数····· 75
- 商环····· 285
- 商群····· 578
- 上下文无关文法····· 643
- 上下文有关文法····· 644
- 设备测试····· 645
- 设备间子系统····· 645
- 设备驱动调度····· 673
- 设计工具····· 646
- 设计规约····· 646
- 设计性语言····· 647
- 射极耦合逻辑····· 732
- 摄像机标定····· 648
- 身份认证协议····· 421
- 砷化镓存储器····· 648
- 砷化镓集成电路····· 649
- 深度优先搜索····· 788
- 神经计算····· 651
- 神经计算机····· 651
- 神经网络芯片····· 652
- 审计过程····· 1017
- 审计尾迹····· 692
- 甚高级语言····· 607
- 甚高速数字用户专用线····· 751
- 甚小口径天线地球站····· 652
- 生成程序····· 12
- 生成元····· 577
- 生物计算····· 653
- 生物特征识别····· 654
- 声纹识别····· 755
- 时变图像····· 807
- 时段演算····· 656
- 时分多路复用····· 657
- 时分多址····· 290
- 时分交换机····· 118
- 时间抽取····· 484
- 时间复杂性····· 659
- 时间片····· 186
- 时空折算关系····· 660
- 时态逻辑····· 660
- 时序系统····· 661
- 实地址····· 919
- 实例法····· 255
- 实例学习····· 254
- 实时仿真····· 502
- 实时绘制····· 662
- 实体安全····· 341
- 实体联系模型····· 662
- 实体联系图····· 663
- 实体完整性····· 706
- 实现语言····· 1067
- 实型····· 419
- 事件调度法····· 495
- 事务····· 505
- 事务处理····· 663
- 事务方式····· 139
- 事务进程监控器····· 663
- 事务元····· 664
- 视觉计算理论····· 664
- 视频光盘机····· 665
- 视频会议····· 666
- 视频随机存取存储器芯片····· 667
- 视图插值····· 668
- 适应谐振理论····· 669
- 收发器····· 670
- 收发器电缆····· 670
- 手持计算机····· 216
- 手势识别····· 670
- 输出设备····· 672
- 输入设备····· 672

- 输入输出管理程序····· 673
- 输入输出技术····· 674
- 输入输出接口····· 674
- 输入输出设备接口····· 674
- 输入输出通道····· 677
- 鼠标器····· 677
- 树····· 679
- 树形索引····· 768
- 数据····· 679
- 数据安全性····· 679
- 数据报····· 189
- 数据编码····· 679
- 数据并行····· 102
- 数据仓库····· 681
- 数据抽象····· 524
- 数据处理速率····· 682
- 数据传输····· 682
- 数据传输差错检测与控制····· 684
- 数据传输速率····· 89
- 数据传送····· 686
- 数据传送率····· 686
- 数据词典管理工具····· 646
- 数据的逻辑独立性····· 695
- 数据的物理独立性····· 695
- 数据电路设备····· 686
- 数据调制····· 687
- 数据定义语言····· 629
- 数据共享····· 688
- 数据划分····· 27
- 数据结构····· 688
- 数据结构化系统开发方法····· 689
- 数据库····· 690
- 数据库管理····· 691
- 数据库管理系统····· 692
- 数据库管理员····· 695
- 数据库机····· 167
- 数据库连通性标准····· 693
- 数据库设计····· 693
- 数据库系统····· 694
- 数据库系统三级结构····· 695
- 数据库性能评价····· 696
- 数据宽度····· 686
- 数据类型····· 696
- 数据链路控制····· 696
- 数据链路控制规程····· 697
- 数据流计算机····· 698
- 数据词典(DD)····· 429
- 数据流图····· 699
- 数据流语言····· 700
- 数据录入设备····· 707
- 数据模式····· 691
- 数据模型····· 700
- 数据驱动····· 698
- 数据融合····· 374
- 数据通路····· 701
- 数据通信····· 701
- 数据通信接口····· 703
- 数据通信接口标准····· 704
- 数据图····· 430
- 数据挖掘····· 706
- 数据完整性····· 706
- 数据依赖····· 707
- 数据帧····· 953
- 数据终端设备····· 707
- 数据准备设备····· 707
- 数理逻辑····· 708
- 数理统计····· 710
- 数论函数····· 997
- 数码相机····· 747
- 数模转换器····· 711
- 数值逼近····· 712
- 数值积分····· 713
- 数值积分法仿真····· 502
- 数值计算····· 715
- 数值计算误差分析····· 717
- 数值微分····· 719
- 数制····· 720
- 数字磁记录····· 722
- 数字磁记录编码方法····· 725
- 数字磁记录检错码····· 728
- 数字磁记录纠错码····· 729
- 数字多用途光盘····· 730
- 数字化仪····· 731
- 数字语音的压缩编码····· 732
- 数字积分机····· 745
- 数字集成电路····· 732
- 数字几何处理····· 735
- 数字计算机····· 736

- 数字可写光盘..... 739
数字可重写光盘..... 740
数字签名..... 741
数字摄像头..... 741
数字视频编辑..... 742
数字视频获取..... 742
数字数据..... 682
数字数据网..... 743
数字图书馆..... 743
数字图像..... 807
数字图像处理..... 744
数字微分分析机..... 745
数字系统逻辑综合技术..... 746
数字系统模拟技术..... 747
数字相机..... 747
数字信号..... 683
数字信号处理器..... 748
数字音频编辑..... 749
数字音频获取..... 749
数字用户专用线..... 750
数组类型..... 751
双极型半导体存储器..... 5
双极型晶体管..... 315
双绞线电缆..... 751
双向耦合器..... 30
水平分片..... 186
水平子系统..... 752
顺序程序设计..... 754
顺序控制..... 754
顺序逻辑程序设计语言..... 1131
说话人识别..... 755
说明..... 756
说明性语言..... 757
私人工作空间..... 880
私人网..... 880
私钥..... 523
私钥密码技术..... 757
私钥密码系统..... 523
死锁..... 758
死锁检测程序..... 47
松散耦合并行处理系统..... 24
搜索估价函数..... 564
素数..... 758
素性测试..... 759
素域..... 993
宿主机..... 423
宿主语言..... 694
算法..... 760
算法动画..... 761
算法设计..... 762
算法学..... 763
算术逻辑部件..... 764
算术逻辑运算..... 766
随机存取存储器..... 1057
随机存取存储器芯片..... 766
随机存取机..... 767
孙子定理..... 768
索引..... 768
- T**
- 台式计算机..... 860
探求因果联系的归纳推理..... 253
特大规模集成电路..... 733
特定应用服务要素..... 852
特权状态..... 91
特征提取..... 770
特征选择..... 770
特征造型技术..... 1008
梯形公式..... 713
体绘制技术..... 770
条件汇编程序..... 286
条件语句..... 985
条码..... 771
条码阅读器..... 771
跳频扩频..... 878
铁电随机存取存储器(FRAM)芯片..... 1035
停-等流控..... 510
停机问题..... 775
通信服务器..... 776
通信控制器..... 776
通信控制设备..... 776
通信顺序进程..... 777
通信系统演算..... 778
通用串行总线接口..... 675
通用多八位编码字符集 ISO/IEC 10646..... 779
通用寄存器..... 780
通用逻辑阵列电路..... 1062
同步并行算法..... 28

同步传输	780
同步光纤网	781
同步控制	1051
同步数据传送	686
同步数字体系	658
同构型表达式	16
冗余	782
同轴电缆	782
统计时分多路复用	659
统一建模语言	783
统一资源定位器	827
图归约机	256
图灵归约	784
图灵机	784
图论	786
图论算法	787
图像	807
图像边界表示	788
图像边缘检测	789
图像编辑程序	12
图像变换	790
图像变换运算	792
图像表示	792
图像并行处理	793
图像处理的基本运算	795
图像的压缩编码	796
图像点运算	798
图像反向滤波复原	799
图像分割	800
图像分析	800
图像复原	800
图像骨架表示	801
图像函数	807
图像获取	801
图像几何特征表示	802
图像几何运算	803
图像矩表示	804
图像卷曲	804
图像理解系统	805
图像邻域运算	806
图像模型	806
图像区域表示	808
图像区域分割	809
图像特征提取	809
图像退化	811
图像维纳滤波复原	812
图像细化	813
图像像素分类	813
图像形态学运算	814
图像旋转	803
图像运动模糊复原	815
图像增强	816
图像重建	794
图像转置	803
图形编辑程序	12
图形变换	817
图形裁剪	817
图形反走样技术	817
图形元文件标准	818
图元生成	818
推理机	819
推理机制	928
推理控制策略	819
吞吐率	393
脱机打印设备	820
脱机设备	820
脱机手写汉字识别	820
W	
外部路由协议	822
外存储设备接口	822
外存储器子系统	824
外键码	422
外联网	826
外排序	548
外围设备接口	674
外延	169
外延数据库	941
外页表	921
外页表项	921
外总线	886
完全偏序	826
完整性	831
万维网	826
万维网数据管理	827
网格编码调制	775
网格计算	827
网关	830

-
- | | | | |
|-----------------|-----|---------------|------|
| 网际协议····· | 830 | 微程序····· | 856 |
| 网际协议安全性协议····· | 926 | 微程序控制器····· | 856 |
| 网卡····· | 845 | 微处理器····· | 857 |
| 网络安全····· | 831 | 微控制器····· | 858 |
| 网络操作系统····· | 832 | 微命令····· | 856 |
| 网络测试····· | 832 | 微内核····· | 860 |
| 网络处理器····· | 833 | 微型计算机····· | 860 |
| 网络服务····· | 834 | 微型驱动器····· | 971 |
| 网络服务质量····· | 835 | 微指令····· | 856 |
| 网络工程····· | 836 | 微组装技术····· | 862 |
| 网络管理····· | 836 | 维护工具····· | 863 |
| 网络管理标准····· | 837 | 维护过程····· | 864 |
| 网络规划····· | 838 | 维也纳开发方法····· | 864 |
| 网络互连····· | 839 | 卫星接入技术····· | 866 |
| 网络互连技术····· | 839 | 卫星通信····· | 866 |
| 网络互连设备····· | 840 | 未格式化容量····· | 88 |
| 网络互连协议····· | 840 | 伪随机数····· | 865 |
| 网络集成服务····· | 841 | 伪向量处理····· | 900 |
| 网络计算模式····· | 841 | 位密度····· | 88 |
| 网络控制协议····· | 115 | 尾数····· | 721 |
| 网络区分服务····· | 841 | 温切斯特盘驱动器····· | 971 |
| 网络入侵检测····· | 842 | 文档····· | 369 |
| 网络软件····· | 842 | 文档分析工具····· | 863 |
| 网络设计····· | 843 | 文档过程····· | 1016 |
| 网络适配器····· | 845 | 文档语言····· | 868 |
| 网络数据单元····· | 845 | 文法····· | 868 |
| 网络数据库····· | 846 | 文化程序设计····· | 869 |
| 网络体系结构····· | 847 | 文件····· | 872 |
| 网络协议····· | 848 | 文件传送····· | 872 |
| 网络协议工程····· | 849 | 文件传送协议····· | 872 |
| 网络协议规范····· | 850 | 文件分配表····· | 1104 |
| 网络协议形式描述技术····· | 850 | 文件服务器····· | 192 |
| 网络协议一致性测试····· | 851 | 文件共享····· | 874 |
| 网络应用服务····· | 852 | 文件管理程序····· | 875 |
| 网络运行环境····· | 852 | 文件逻辑结构····· | 875 |
| 网络中间件····· | 853 | 文件目录····· | 1103 |
| 网络资源预约协议····· | 853 | 文件物理结构····· | 875 |
| 网络最大传送单元····· | 830 | 文语转换····· | 875 |
| 网桥····· | 854 | 纹理生成····· | 876 |
| 网状模型····· | 855 | 纹理映射····· | 876 |
| 网状数据库····· | 855 | 问题对象····· | 138 |
| 微波····· | 866 | 问题解决过程····· | 1017 |
| 微波通信····· | 856 | 握手式通信····· | 778 |
| 微操作····· | 856 | 无编号帧····· | 441 |

无焊压接	878
无线局域网	878
无线应用协议	880
无向边	786
无向图	882
无源元件	315
无约束最优化方法	1091
吴方法	877
物理地址	92
物理设计	694

X

系统测试	46
系统程序设计语言	883
系统兼容性	883
系统维护	884
系统性能指标	885
系统总线	886
系统总线控制器	1082
系统总线仲裁器	140
细胞自动机	887
狭义模糊逻辑	530
下推自动机	888
先行进位加法器	1053
显示卡	891
显示器	889
显示适配器	889
现场总线控制系统	892
线程	893
线路单元	777
线性代数方程组数值解法	894
线性电源	1029
线性多步法仿真	503
线性逻辑	897
线性文法	898
线性有界自动机	898
线性纵向记录	278
陷阱	1040
详细设计工具	646
相变存储器(OUM)芯片	1035
相变光盘驱动器	899
相对误差	718
相空间语义	897
相似性和对偶性原理	196
相位编码	726
相位调制	687
相移键控	774
相移键控法	687
箱体式集线器	334
响应时间	393
向后误差分析法	718
向量程序设计	29
向量计算	900
向前误差分析法	718
项目管理工具	900
象集	968
消去法	894
消息包	902
消息处理系统	901
消息传递	902
消息传递并行程序设计	29
消隐技术	903
小规模集成电路	732
小脑网络模型	904
小型计算机	905
协处理器	905
协同例程	906
协议转换器	840
协作策略	185
辛普森公式	713
新闻组	1140
新一代网际协议	907
信念	908
信息	909
信息安全	341
信息检索方法	909
信息检索中的相关反馈	909
信息交换用汉字编码字符集 GB 2312—1980	
.....	910
信息交换用汉字编码字符集的扩充 GB 18030	
.....	911
信息系统安全	341
信息系统安全评估准则	912
信息隐蔽	912
信息帧	441
信元	955
信元交换	912
信噪比	82

行波加法器	1052	言语(语音)合成器	934
形式方法	912	言语(语音)识别的特征抽取	935
形式功能规约	914	言语(语音)识别中的语言模型	936
形式规约	914	颜色复制	936
形式设计规约	916	颜色管理	937
形式语言理论	916	颜色模型	938
形式语义	919	掩模型只读存储器芯片	1033
性能价格比	393	演化策略	997
虚地址	919	演化模型	940
虚电路	188	演绎数据库	941
虚拟存储器	919	演绎推理	819
虚拟机	921	演绎综合方法	941
虚拟现实	922	验证过程	1016
虚拟现实输出设备	923	样条函数	942
虚拟现实输入设备	924	遥感信息处理	943
虚拟终端	925	页表	93
虚拟专网	926	页表项	920
虚拟子网	837	页框	93
虚通路	955	页面	93
虚信道	956	页面描述语言	944
需求定义语言	926	页面替换	93
需求分析工具	928	页式存储系统	99
需求工程	929	液晶显示器	891
需求驱动	256	一次可擦可编程只读存储器芯片	468
序列密码	757	一阶逻辑	945
序数	929	一致逼近	712
序型	929	依存语法	1077
选择结构	260	移动式计算机	947
选择排序	549	移动数据库	948
选择通道	677	移动通信	949
学习计算理论	930	移动通信网	949
寻道时间	88	移码	722
寻址方式	931	移位寄存器	735
循环调度	20	遗传算法	949
循环结构	260	遗传学习	952
循环卷积	484	以太网	953
循环群	577	异步并行算法	28
循环冗余检错	684	异步传输	955
训练仿真器	932	异步传送模式	955
训练样本集	536	异步传送模式局域网	957
Y			
雅可比方法	448	异步控制	1051
言语(语音)合成方法	934	异步数据传送	686
		异常处理	958
		异或	517

- 译后编辑..... 959
- 译码..... 596
- 译前编辑..... 959
- 易失性存储器芯片..... 6
- 因特网服务器..... 192
- 因子分解..... 959
- 阴极射线管..... 891
- 引用透明..... 262
- 引址调用..... 754
- 隐函数曲面..... 959
- 印前图像处理技术..... 960
- 印刷体汉字识别..... 961
- 印刷文本版面分析..... 961
- 印制板..... 967
- 印制板测试..... 962
- 印制板模拟在线测试..... 965
- 印制板设计..... 963
- 印制板数字在线测试..... 965
- 印制板在线测试..... 965
- 印制板制造..... 967
- 印制板综合测试仪..... 963
- 映射..... 968
- 硬磁盘存储器..... 968
- 硬磁盘控制器..... 970
- 硬磁盘驱动器..... 971
- 硬磁盘驱动器测试..... 972
- 硬磁盘适配器..... 973
- 硬件描述语言..... 973
- 硬件同步机制..... 974
- 硬件验证..... 976
- 硬连线控制器..... 976
- 永久虚电路..... 189
- 用户工程师盘..... 772
- 用户界面..... 976
- 用户界面管理系统..... 977
- 用户数据报..... 978
- 用户数据报协议..... 978
- 优先数调度..... 593
- 有补格..... 214
- 有限域..... 993
- 有限元方法..... 979
- 有限自动机..... 981
- 有向边..... 786
- 有向图..... 983
- 有效频带..... 683
- 有效数据传送率..... 686
- 有效数字..... 718
- 有源元件..... 315
- 右陪集..... 577
- 余3码..... 720
- 与..... 517
- 语法..... 983
- 语法分析..... 983
- 语法分析程序的生成程序..... 14
- 语法图..... 984
- 语境分析..... 984
- 语句..... 985
- 语料库..... 986
- 语料库语言学..... 988
- 语言处理系统..... 989
- 语言结构..... 927
- 语义..... 991
- 语义分析..... 991
- 语义网络表示..... 992
- 语音合成器..... 934
- 语用..... 992
- 语用分析..... 927
- 域..... 992
- 域间路由协议..... 822
- 域名服务器..... 994
- 域名系统..... 993
- 域完整性..... 706
- 元规则..... 53
- 元器件测试..... 994
- 元器件可靠性..... 995
- 元器件老化..... 995
- 元器件筛选..... 996
- 元数据..... 146
- 元数据库..... 681
- 元素..... 808
- 元语言..... 996
- 原码..... 721
- 原码加减交替法..... 76
- 原始递归函数..... 997
- 原象集..... 968
- 原型速成方法..... 997
- 原语..... 18
- 原子操作..... 974

源程序	13
源程序文档与结构图生成工具	498
源代码调试程序	772
远程过程调用	478
远程教育	998
远程网	250
远程网络监控	999
远程医疗	1000
远程终端	1055
约瑟夫逊结器件	57
约束传播	1002
约束推理	1002
约束最优化方法	1092
运动分析	1003
运动图像	807
运动图像压缩编码标准	1003
运算器	1005
运算速度评价	1005
运行安全	341
运行时刻	990
运作过程	1006

Z

载波监听多址访问-冲突检测	954
载波扩展	566
再次工程工具	863
在线攻击	421
造型技术	1008
噪声	82
增强现实	1008
栈指示器	780
栈自动机	1009
锗硅异质结器件	1009
阵列处理机	1012
针床夹具	966
针式打印机	295
真彩色	891
真实感图形生成	1011
真值表	517
真子集	328
整流电源	1029
整型	419
整字形打印设备	295

正规子群	578
正交调幅	775
正文编辑程序	12
正则表达式	1012
正则文法	1013
证书权威机构	422
帧	697
帧猝发	566
帧存储器芯片	667
帧中继交换	1015
帧中继数据网	1016
支持过程	1016
知识表示	1017
知识工程	1020
知识获取	1021
知识获取工具	1023
知识精化	1025
知识库	1026
知识库机	167
执行周期	662
直接表示	1027
直接存储器存取	1028
直接数字控制	358
直接序列扩频	878
直觉主义逻辑	1028
直流电源	1029
值	1032
值域	968
只读存储器	1032
只读存储器芯片	1032
只读光盘驱动器	1035
指称语义	1036
指挥控制通信情报系统	1037
指令	1041
指令格式	1037
指令级并行处理	1038
指令计数器	64
指令寄存器	1039
指令类型	1039
指令系统	1041
指令周期	661
指纹识别	1042
指针对象	1043
指针类型	1043

- 质量保证过程 1016
- 智能机器人 1043
- 智能集线器 334
- 智能卡阅读器 1045
- 智能设备接口 675
- 置标语言 1046
- 置换群 578
- 中断 1047
- 中规模集成电路 732
- 中国邮路问题 1047
- 中继器 1048
- 中间件 184
- 中日韩统一汉字 1049
- 中文信息处理 1050
- 中文信息检索 1050
- 中央处理器 1051
- 终端 1055
- 终端服务器 1055
- 终端设备 1055
- 重复语句 985
- 重构 594
- 重写规则法 73
- 周期窃取 1028
- 逐步求精 1068
- 主存储器 1057
- 主索引 768
- 主页 1057
- 属性文法 678
- 柱面 88
- 专家系统 1058
- 专家系统开发环境 1060
- 专家系统开发语言 1060
- 专家系统外壳 1060
- 专用集成电路 1061
- 专用逻辑集成电路 1061
- 专用小交换机 1063
- 转发器 840
- 转换风范 611
- 转换检测缓冲器 1064
- 转移指令 64
- 装入程序 1064
- 状态转移图 1064
- 桌面出版系统 130
- 资源共享模式 1065
- 子程序 1065
- 子程序说明 756
- 子例程 1066
- 子群 577
- 子图 786
- 子域 993
- 自编译程序 1067
- 自底向上方法 1068
- 自顶向下方法 1068
- 自动编译器 989
- 自动标引 1069
- 自动盒带库 1070
- 自动机理论 1071
- 自动生成 635
- 自动推理 1072
- 自动验证 635
- 自动增减量寄存器 780
- 自检验电路 1073
- 自然景物造型 1073
- 自然演绎法 1074
- 自然语言处理 1074
- 自然语言处理的词法分析 1075
- 自然语言处理的句法分析 1076
- 自然语言理解 1077
- 自同步编码 680
- 自由软件 1118
- 自展 1067
- 自治系统(AS) 513
- 自组织映射模型 1078
- 字 737
- 字长 737
- 字符串类型 751
- 字符集 1066
- 字符型 419
- 字节 737
- 字节多路通道 677
- 纵向磁记录 724
- 纵向转换 72
- 总线标准 1081
- 总线带宽 887
- 总线控制器 1082
- 总线仲裁器 1082
- 综合理论性能 1079

综合业务数字网	1079
租用线路网	1083
组播	250
组合测试	46
组合逻辑	1084
组合逻辑系统	1084
组合算法	1084
组合索引	768
组合学	1085
组合子	1084
最大公因子	1088
最大流	1088
最短路径问题	1089
最坏情况复杂性	196

A

A* 算法	1095
Ada 语言	1096
AIT 格式盒带	279
AI 工具包	1060
ALGOL 60 语言	1097
ALGOL 68 语言	1097
Amdahl 定律	392
AO* 算法	1098
ASCII 码	737

B

BASIC 语言	1099
BCD 码	720
BCY 语言	1100

C

C++ 语言	1100
CD- G	665
CD- I	665
CD- V	665
CGI 图形接口标准	1101
CIP-L 语言	1101

最弱前置条件	1089
最弱前置条件方法	1089
最弱前置条件演算	915
最小二乘法	1089
最小生成树	1090
最小势能原理	979
最优化方法	1091
左陪集	577
作业	1093
作业调度	1093
作业管理程序	1093
作业控制	1093
作业流	1093
坐标映射和重采样	803

COBOL 语言	1101
----------------	------

C 语言	1102
------------	------

D

DOS 操作系统	1103
D 触发器	733

E

EIA 接口	704
EIDE 接口	969
Eiffel 语言	1105
EPP 接口	676

F

FGSPEC 语言	1106
FORTTRAN 语言	1107
FP 语言	1107

G

GKS 图形标准	1108
Gopher 服务	1108
Gouraud 明暗处理	1109
Groebner 基方法	377
GSPEC 语言	1109

H

- Hopfield 神经网络模型** 1110
- Horn 子句** 1131

I

- I/O 总线** 139
- IBM PC/XT 总线** 1081
- IC 卡** 1046
- IDE 接口** 969
- IEEE 802** 444
- IEEE 802.10** 444
- IEEE 802.11** 444
- IEEE 802.12** 445
- IEEE 802.14** 445
- IEEE 802.15** 445
- IEEE 802.16** 445
- IEEE 802.2** 444
- IEEE 802.3** 444
- IEEE 802.4** 444
- IEEE 802.5** 444
- IEEE 802.6** 444
- IEEE 802.9** 444
- IEEE-1394 接口** 675
- Internet(因特网)服务提供者** 1112
- Internet** 1111
- Internet 地址** 1112
- Internet 基本服务** 1113
- Internet 控制报文协议** 831
- Internet 体系结构** 1114
- IP 因特网** 1111
- ISO 9314** 445

J

- Jackson 系统开发方法** 1115
- Java 语言** 1116
- J-K 触发器** 733

K

- Kerberos 鉴别** 1117

L

- Larch 语言** 1117

- LBA 问题** 917
- LCP 包** 115
- LINPACK 基准程序** 1118
- Linux 操作系统** 1118
- LISP 语言** 1119
- Listserv 论坛** 1120
- LIW 处理机** 1141
- LR(*k*) 文法** 1120

M

- Miranda 语言** 1120
- ML 语言** 1121
- Modula-2 语言** 1121

N

- NP 类问题** 1121
- NP 完全问题** 1122
- NP 完全问题近似方法** 1123
- NP 完全性理论** 1123
- n* 次样条函数** 942

O

- OBJ 语言** 1125
- Occam 语言** 1125
- OpenGL 图形标准** 1126
- OSI 管理体系结构** 1126
- OSI 基准(参考)模型** 461
- OSPF 协议** 1127

P

- PARLOG 语言** 1128
- PASCAL 语言** 1129
- PDL 语言** 1129
- PHIGS 图形标准** 1130
- Phong 明暗处理** 1131
- PPP 帧结构** 115
- PROLOG 语言** 1131
- PSA 程序** 1131
- PSL 语言** 1132
- PV 操作** 18
- P 类问题** 1132

Q

- QR 方法** 448

- | | |
|---|---|
| <p style="text-align: center;">R</p> <p>RIP 协议 1132</p> <p>RS232C 704</p> <p>RS449 705</p> <p>R-S 触发器 733</p> <p style="text-align: center;">S</p> <p>SCSI 接口 969</p> <p>Seeheim 模型 978</p> <p>SETL 语言 1133</p> <p>Smalltalk 语言 1133</p> <p>SMDS 接口协议(SIP) 426</p> <p>SNOBOL 语言 1134</p> <p>SPEC 基准程序 1135</p> <p style="text-align: center;">T</p> <p>TCP/IP 互联网 1114</p> <p>TCP/IP 协议集 1135</p> <p>TPC 基准程序 1137</p> <p>T 触发器 733</p> <p style="text-align: center;">U</p> <p>Unicode 编码字符集 1137</p> <p>UNIX 操作系统 1138</p> <p>USENET 新闻 1139</p> <p style="text-align: center;">V</p> <p>V.24 705</p> <p>V.35 705</p> <p>VAL 语言 1140</p> <p>VCD 665</p> <p>VLW 处理机 1140</p> <p>VLSI 算法 1141</p> <p style="text-align: center;">W</p> <p>Warnier-Ort 方法 689</p> <p>Warnier 图 689</p> <p>Warren 抽象机 517</p> <p>Windows 操作系统 1143</p> <p style="text-align: center;">X</p> <p>X.21 705</p> <p>X.25 建议 189</p> | <p>X.400 系列建议 901</p> <p>X.75 建议 1145</p> <p>XCY 语言 1146</p> <p>XYZ/E 语言族 1146</p> <p style="text-align: center;">Z</p> <p>Z 语言 1146</p> <p style="text-align: center;">α</p> <p>α-β 剪枝 1147</p> <p>α 测试 46</p> <p style="text-align: center;">β</p> <p>β 测试 46</p> <p style="text-align: center;">λ</p> <p>λ 演算 1148</p> <p style="text-align: center;">π</p> <p>π 演算 1150</p> <p style="text-align: center;">ω</p> <p>ω-有限自动机 1151</p> <p style="text-align: center;">Σ</p> <p>Σ (基调) 代数 1152</p> <p style="text-align: center;">0</p> <p>0 型文法 1154</p> <p style="text-align: center;">1</p> <p>1 型文法 1154</p> <p style="text-align: center;">2</p> <p>2 型文法 1154</p> <p style="text-align: center;">3</p> <p>3 型文法 1154</p> <p style="text-align: center;">4</p> <p>4 mm 数字音频盒带 279</p> <p style="text-align: center;">8</p> <p>8 mm 视频盒带 279</p> |
|---|---|

附录 I 缩略语

缩略语	英文全名	中文译名
AAD	analog alignment diskette	模拟校准盘
AAL	ATM adaptive layer	ATM 适配层
ABI	application binary interface	应用二进制接口
ABM	asynchronous balanced mode	异步平衡模式
ABR	alternate bit rate	自适应式比特率(服务)
ABS	Agfa balanced screening	Agfa[公司]均衡挂网(技术)
AbS	analysis-by-synthesis	合成-分析法
AC	access control	访问控制
ACID	atomicity, consistency, isolation and durability	不可再分割性, 一致性, 隔离性与耐用性
ACL	agent communication language	主体通信语言
ACL	Association for Computational Linguistics	计算语言学学会(美国)
ACM	Association for Computing Machinery	计算机协会(美国)
ACSE	association control service element	联系控制服务要素
ACSL	advanced continuous simulator language	高级连续仿真语言
ADC	analog-to-digital converter	模数转换器
ADL	architecture description language	体系结构描述语言
ADM	add/drop multiplexer	多路复用与多路分解器
ADPCM	adaptive differential pulse code modulation	自适应差分脉[冲编]码调制
ADSL	asymmetric digital subscriber line	非对称数字用户线
AES	application environment specification	应用环境规范
AET	active edge table	有效边表
AF	assured forwarding	保证转发
AFS	Andrew File System	(卡内基-梅隆大学的)分布式文件系统
AFS	automatic file system	自动文件系统
AGC	automatic gain control	自动增益控制
AH	authentication leader	鉴别报头
AHPL	a hardware programming language	硬件程序设计语言, AHPL 语言
AI	artificial intelligence	人工智能

缩略语	英文全名	中文译名
AIT	advanced intelligent tape	高级智能磁带(技术)
AIX	advanced interactive executive	高级交互执行程序
ALGOL	algorithmic language	算法语言
ALICE	automated location of isolation and continuity errors	孤立与连续性错误的自动定位
ALOA	asset library open architecture framework	开放体系结构的构件库框架
ALPAC	Automatic Language Processing Advisory Committee	语言自动处理咨询委员会
ALU	arithmetic and logic unit	算术逻辑部件, 运算器
AM	address mark	地址标志
AMA	abstract muscle action	抽象肌肉动作
AMBIT	Acronym May Be Ignored Totally	AMBIT(语言)
AMD	Advanced Micro Devices Inc.	先进微型器件公司, AMD 公司
AME	super metal evaporation	高级金属蒸镀带(技术)
AMF	address mark found	找到地址标志
AMP	advanced metal evaporation	高级金属粉末
ANN	artificial neural network	人工神经网络
ANS	Advanced Network Services, Inc.	先进网络服务公司
ANSA	Advanced Network System Architecture	先进网络体系结构(日本东芝公司)
ANSI	American National Standards Institute	美国国家标准学会
AOP	agent-oriented programming	面向主体的程序设计
AP	access point	访问点
AP	array processor	阵列处理机
API	application programming interface	应用程序设计接口
APL	A Programming Language	APL[程序设计]语言
APON	ATM passive optical network	异步传输模式无源光纤网
APPANET	Advanced Research Project Agency Network	(美国) APPA 网
APT	Automatically Programmed Tools	自动程控机床(语言), APT(语言)
AR	augmented reality	增强现实
ARGUS	automatic routine generating and updating system	自动例程生成和更新系统
ARM	asynchronous response mode	异步响应模式
ARP	address resolution protocol	地址解析协议
ARPA	Advanced Research Projects Agency	(美国)高级研究计划署
ARQ	automatic repeat request	停-等自动重复请求
ART	adaptive resonance theory	自适应谐振理论
AS	activity scanning	活动扫描(法)
AS	autonomous system	自治系统
ASA	Advanced Software Automation	高级软件自动化(公司)

缩略语	英文全名	中文译名
ASC	American Standards Committee	美国标准委员会
ASCI	Accelerated Strategic Computing Initiative	战略的加速计算倡导者联合会
ASCII	American standard code for information interchange	美国信息交换标准[代]码,ASCII 码
ASIC	application specific integrated circuit	专用集成电路
ASIC	application specified integrated circuit	按应用需求定制的集成电路
ASK	amplitude shift keying	幅移键控
ASK	a simple knowledgeable system	一种简单知识系统
ASL	American sign language	美国符号语言
ASLIC	application specific logic integrated circuit	专用逻辑集成电路
ASN. 1	abstract syntax notation. one	抽象句法表示法 1
ASP	application service provider	应用服务提供商
ASPOL	a simulation process-oriented language	一种面向模拟过程语言
ASR	automated send/receive	自动发送接收
ATA	advanced technology attachment	高级技术配件
ATAPI	advanced technology attachment packet interface	高技术配件分组接口
ATC	address translation cache	地址转换高速缓存
ATC	air traffic control	空中交通指挥
ATE	automatic test equipment	自动测试设备
ATM	asynchronous transfer mode	异步传送模式
ATM	automatic teller machine	自动出纳机
ATMLAN	asynchronous transfer mode local area network	异步传送模式局域网
ATN	augmented transition network	扩充转移网络,增强型转移网络
ATP	automatic theorem proving	自动定理证明
AU	access unit	访问单元
AUC	authentication center	鉴权中心
AUI	attachment unit interface	连接部件接口
AURA	automated reasoning assistant	自动推理助理(系统),AURA(系统)
AVS	application visualization system	应用可视化系统
BAF	branch address field	转移地址字段
BAS	building automatic system	建筑物自动化系统
BASIC	Beginner's All-purpose Symbolic Instruction Code	初学者通用符号指令码(语言), BASIC(语言)
BBN	Bolt, Beranck and Newman, Inc.	博尔特·贝拉尼克和纽曼公司
BBS	bulletin board system	公告板系统
B2C	business to consumer	商家到客户
BCD	binary-coded decimal	二-十进制

缩略语	英文全名	中文译名
BCF	branch control field	转移控制字段
BCH	Bose-Chaudhuri-Hocqueghem (code)	博斯-乔赫里-霍克文黑姆(码), BCH(码)
BCPL	basic combined programming language	基本的组合式程序设计语言
BDI	belief-desire-intention	信念-期望-意图
BDI	business data interchange	商务数据交换
BECN	backward explicit congestion notification	后向显示拥塞控制标志
Bellcore	Bell Communication Research	贝尔通信研究所
BFL	buffered field effect transistor logic	缓冲场效应晶体管逻辑电路
BGA	ball grid array package	球栅阵列封装
BG	black generation	黑色生成函数
BGP	Border Gateway Protocol	边界网关协议
BHM	basic hypermedia model	基本超媒体模型
BICFET	bipolar inversion channel field effect transistor	双极反型沟道场效应晶体管
BIDM	basic interoperability data model	基本互操作数据模型
BiFET	bipolar field effect transistor	双极型场效应晶体管
BiMOS	bipolar MOS	双极型金属-氧化物-半导体
BIOS	basic input/output system	基本输入输出系统(管理程序)
B-ISDN	broadband integrated services digital network	宽带综合业务数字网, 宽带 ISDN
BIST	built-in self-test	内装自测试, 内建自测试
BLAS	basic linear algebra subprogram	基本线性代数子程序
BLISS	basic language for the implementation of system software	实现系统软件的基本语言
BLOB	binary large object	二元大客体
BMP	basic multilingual plane	基本多文种平面
BNA	Burroughs Network Architecture	宝来网络体系结构(美国 Burroughs 公司)
BNF	Backus-Naur form alism	巴克斯-诺尔形式体系
BNF	Backus normal form	巴克斯范式
BP	back propagation	BP 算法, 反传
BPF	band pass filter	带通滤波器
BPR	business process reengineering	企业流程重组
BPSK	binary phase shift keying	二进制相移键控
B-rep	boundary representation	边界表示法
BRI	basic-rate interface	基本速率接口
BS	base stations	基站

缩略语	英文全名	中文译名
BSC	base station control	基站控制器
BSD	Berkeley software distribution	(美国加州)伯克利软件发行版本
BSP	board support package	板支持组件
BSP	Burroughs Scientific Processor	宝来(公司)科学处理机
BSS	base station subsystem	基站子系统
BTB	branch target buffer	转移目标缓冲
BTC	base transceiver stations	基站收发信机
CA	certificate authority	证书权威(机构)
CACI	Consolidated Analysis Centers, Inc.	联合分析中心
CACM	Communications of the Association for Computing Machinery	(美国)计算机协会通信
CADAM	computer-graphics-augmented design and manufacturing	计算机图形扩充设计与制造
CAD/CAM	computer-aided design/computer-aided manufacturing	计算机辅助设计和制造
CAD	computer-aided design	计算机辅助设计
CADDS	computer-aided design and drafting system	计算机辅助设计与制图系统
CAE	computer-aided engineering	计算机辅助工程
CAFS	content addressable file storage	按内容寻址文件存储器
CAGD	computer-aided geometric design	计算机辅助几何设计
CAI	computer-assisted instruction	计算机辅助教学
CALS	commercial at light speed	光速商务
CAM	cellular automata machine	细胞自动机
CAM	computer-aided manufacturing	计算机辅助制造
CAM	computer-aided mapping	计算机辅助地图绘制
CAMI	automated process planning system	自动工艺规划系统
CAPP	computer-aided process planning	计算机辅助工艺规划(系统)
CAQ	computer-aided quality control	计算机辅助质量控制
CASE	common application service element	公共应用服务要素
CASE	computer-aided software engineering	计算机辅助软件工程
CAT	computer-aided testing	计算机辅助测试
CAT	computer-aided test	计算机辅助检测
CATIA	computer-graphics-aided three-dimensional interactive application	计算机图形辅助三维交互应用
CATV	cable television	有线电视
CAV	constant angular velocity	恒角速度
CAVE	computer automatic virtual environment	洞穴式显示装置(计算机自动虚拟环境)

缩略语	英文全名	中文译名
CBE	computer-based education	计算机辅助教育
CBL	computer-based learning	计算机辅助学习
CBR	case-based reasoning	基于案例的推理
CBR	constant bit rate	恒定比特率(服务)
CBSD	component-based software development method	基于构件的软件开发方法
CBSE	component-based software engineering	基于构件的软件工程
CBT	computer-based teaching	计算机辅助教学
CBX	computerized branch exchange	计算机化小交换机
CCA	common cryptography architecture	公共密码体系[结构]
CCA	Computer Corporation of America	美国计算机公司
CC	common criteria of IT security evaluation	通用安全评估准则
CCD	charge coupled device	电荷耦合器件
CC-DOS	Chinese Character Disk Operating System	汉字磁盘操作系统
CCFL	charge-coupled FET logic	电荷耦合场效应晶体管逻辑电路
CCIR	Consultative Committee on International Radio	国际无线电咨询委员会
CCITT	Consultative Committee on International Telegraph and Telephone	国际电话电报咨询委员会
CC-NUMA	cache coherency non-uniform memory access	高速缓冲存储器一致性非均匀存储器访问
CCS	calculus of communicating system	通信系统演算(语言), CCS(语言)
CCSS	common channel signalling system	公共信道信令系统
CCTA	Central Computer and Telecommunications Agency	中央计算机和电信总局(英国)
CCU	communications control unit	通信控制器
CDDI	copper distributed data interface	铜线分布式接口
CDE	common desktop environment	公共桌面环境
CD-G	CD-graph	CD-G 视盘
CD-I	CD-interactive	交互式激光视盘
CDI	compact disk interactive	交互式光盘
CDL	component definition language	构件定义语言
CDL	computer design language	计算机设计语言
CDMA	code division multiple access	码分多址接入
CD-R	compact disc-recordable	数字可写光盘
CD-ROM	compact disc-read only memory	只读光盘, CD-ROM 盘
CD-RW	compact disc-rewritable	数字可重写光盘
CD-V	CD-video	CD-V 视盘
CEC	Commision of the European Communities	欧洲共同体委员会

缩略语	英文全名	中文译名
CE	concurrent engineering	并行工程
CE	customer engineer	现场工程师
CEDAR	computer-aided environmental design, analysis and realization	计算机辅助环境设计,分析和实现
CEGA	Chinese enhanced graphics adapter	汉字 EGA 图形适配器
CEL	current events list	当前事件表
CELP	code excited linear prediction coding	码激励线性预测编码
CELP	code excited linear predictive	码激励线性预测法
CEPT	Conference of European Postal and Telecommunications Administrations	欧洲邮政电信管理会议
CF	certainty factor	确信度
CF	compact flash	袖珍闪存
CFG	context-free grammar	上下文无关文法, 2 型文法
CFL	context-free language	上下文无关语言
CFS	computer file system	计算机文件系统
CGA	colour graphics adapter	彩色图形适配器
CG	computer graphics	计算机图形学
CGI	common gateway interface	公共网关接口
CGI	Computer Graphics Interface Standard	CGI 图形接口标准
CGM	Computer Graphics Metafile Standard	图形元文件标准
CGRM	Computer Graphics Reference Model	计算机图形基准模型
CIC	Computer Information Center Ltd.	计算机信息中心有限公司, CIC 公司
C3I	Command, Control, Communication and Intelligence	指挥、控制、通信及情报系统
C4I	Command, Control, Communication, Computer and Intelligence	指挥、控制、通信、计算机及情报
CICS	client information control system	客户信息控制系统
CIF	common intermediate format	公用中分辨率图像格式
CIFS	common internet file system	通用互连文件系统
CIM	computer integrated manufacturing	计算机集成制造
CIMS	computer-integrated manufacturing system	计算机集成制造系统
CIP	Chinese information processing	中文信息处理
CIP	computer-aided, intuition-guided programming	计算机辅助-直觉指导的程序设计
CIRC	cross interleave Reed-Solomon code	交叉交错里德-索罗门码
CISC	complex instruction set computer	复杂指令集计算机
C4ISR	Command, Control, Communication, Computer, Intelligence, Surveillance and Reconnaissance	指挥、控制、通信、计算机、情报、监视及侦察
CITIS	contractor integrated technical information service	主承包商技术信息集成服务

缩略语	英文全名	中文译名
CIX	commercial Internet exchange	商用 Internet 交换
CJK	China-Japan-Korea	中日韩
CLA	carry-lookahead adder	先行进位加法器
CLFL	complement level field effect transistor	互补电平场效应晶体管
CLIPS	C language integrated production system	C 语言综合生产系统
CLNP	connectionless-mode network protocol	无连接方式网络协议
CLNS	connectionless-mode network service	无连接方式网络服务
CLOS	CLOSE statement processor	CLOSE 语句处理程序
CLP	cell loss priority	信元丢失优先级
CLP	constraint logic programming	约束逻辑程序设计(语言)
CLS	conceptual learning system	概念学习系统
CLV	constant linear velocity	恒线速度
CMAC	cerebellar model articulation controller	小脑网络模型
CMAR	control memory address register	控存地址寄存器
CM	control memory	控制存储器,控存
CMI	computer-managed instruction	计算机管理教学
CMIP	Common Management Information Protocol	公共管理信息协议
CMIS	common management information service	公共管理信息服务
CMISE	common management information service element	公共管理信息服务要素
CML	Chemical Markup Language	化学置标语言
CMM	capability maturity model	能力成熟度模型
CMMI	Capability Maturity Model Integration	能力成熟度模型集成(项目)
CMOS	complementary metal-oxide-semiconductor	互补金属氧化物半导体
CMOT	common management information service and protocol on TCP/IP	公共管理信息服务和基于 TCP/IP 的协议
CMP	chemical mechanical polish	化学机械抛光
CMY	cyan-magenta-yellow	青-品红-黄(模型)
CMYK	cyan-magenta-yellow-black	青-品红-黄-黑(模型)
CNC	computer numerical control	计算机数值控制,计算机数控
COB	chip on board	板上芯片(封装)
COBOL	common business-oriented language	面向商业的通用语言
COBUILD	COLLINS Birmingham University International Language Database	COLLINS 伯明翰大学国际语言数据库
CO-CAD	cooperative computer aid design	协同计算机辅助设计
CO	central office	中心局
CODASYL	Conference on Data System Languages	数据系统语言研究会

缩略语	英文全名	中文译名
COMA	cache-only memory architecture	惟高速缓存存储体系结构
COM	Component Object Model	(微软)构件对象模型
COMIT	compiler of Massachusetts Institute of Technology	麻省理工学院编译程序(语言)
CORBA	Common Object Request Broker Architecture	公共对象请求代理体系结构
COS	cooperation for open systems	开放系统协作
COSE	Common Open Software Environment	公共开放软件环境(组织)
COSE	common open system environment	通用开放系统
COTS	commercial-off-the-shelf	货架商品式(构件)
COTS	commodity off the shelf	流行商品
CPE	customer premises equipment	用户宅院设备
CPI	cycles per instruction	执行每条指令的平均周期数
CPM	critical path method	关键路径方法
CPO	complete partial order	完全偏序
CPU	central processing unit	中央处理机,中央处理器
CRC	cyclic redundancy check	循环冗余检验
CRCW	concurrent read concurrent write	并发读写
CREW	concurrent read exclusive write	并发读互斥写
CRL	certificate revoke list	证书撤销列表
CRT	cathode ray tube	阴极射线管
CRTC	cathode ray tube controller	阴极射线管控制器,CRT 控制器
CS	convergence sublayer	会聚子层
CSA	carry save adder	保留进位加法器
CSCW	computer-supported cooperative work	计算机支持的协同工作,群体计算
CSG	constructive solid geometry	构造的实体几何
CSG	context sensitive grammar	上下文有关文法
CSL	context sensitive language	上下文有关语言
CSLI	Center for the Study of Language and Information	语言和信息研究中心(美国 Stanford 大学)
CSMA/CD	carrier sense multiple access with collision detection	带碰撞检测的载波侦听多址访问
CSM	command service module	命令服务模块
CSP	chip scale package	芯片尺寸封装
CSP	Communication Sequential Processing	通信顺序处理(语言),CSP(语言)
CSP	communicating sequential process	通信顺序进程
CSP	cross system product	交互系统产品
CSS	cascading style sheets	级联样式表

缩略语	英文全名	中文译名
CSS	chunk self-scheduling	块[自]调度
CSS	computer system simulator	计算机系统模拟器
CSS	contact start stop	接触启停(技术)
CSSL	continuous system simulation language	连续系统仿真语言
CSU	channel service unit	信道服务部件
CT	computer tomography	计算机断层摄影
CTP	composite theoretical performance	综合理论性能
C-T-R	Calculating, Tabulating and Recording Co.	计算、制表与录制公司, C-T-R 公司
CVC	call virtual circuit	呼叫虚电路
CVD	chemical vapor deposition	化学气相淀积
CVS	concurrent versions system	并行(软件)版本控制系统
CWA	closed-world assumption	封闭世界假设
DA	design automation	设计自动化
DA	destination address	目的地址
DAC	digital-to-analog converter	数模转换器
DAISY	dynamically architected instruction set from Yorktown	Yorktown 动态构造指令集
DAM	direct access memory	直接存取存储器
DARPA	Defense Advanced Research Projects Agency	国防高级研究计划署(美国防部)
DAS	direct-attached storage	直接连接存储
DAT	digital audio tape	数字音频磁带
DB	database	数据库
DBA	database administrator	数据库管理员
DBCLI	Database Call Level Interface	数据库调用级接口(草案)
DBCS	double byte character sets	双字节字符集
DBMS	database management system	数据库管理系统
DBS	direct broadcast systems	卫星直播系统
DBTG	Data Base Task Group	数据库任务组
DC	device coordinate system	设备坐标系
DCE	data circuit equipment	数据电路设备
DCE	distributed computing environment	分布式计算环境
DCFL	direct coupled field effect transistor logic	直接耦合场效应晶体管逻辑电路
DCG	definite clause grammar	定子句语法
DCH	dependence convex hull	依赖凸边域(方法)
DCNA	data communication network architecture	数据通信网络体系结构
DCOM	distributed component object model	分布式构件对象模型
DCS	distributed control system	集散控制系统

缩略语	英文全名	中文译名
DCSL	deterministic context sensitive language	确定型上下文有关语言
DCT	discrete cosine transform	离散余弦变换
DD	data dictionary	数据词典
DDA	digital differential analyzer	数字微分分析仪
DDBMS	distributed data base management system	分布式数据库管理系统
DDC	direct digital control	直接数字控制
DDD	digital diagnostic diskette	数字诊断盘
DDE	dynamic data exchange	动态数据交换
DDI	data display indicator	数据显示指示器
DDI	direct digital interface	直接数字接口
DDL	data definition language	数据定义语言
DDL	digital system design language	数字系统设计语言
DDL	domain description language	领域描述语言
DDMS	distributed database management system	分布式数据库管理系统
DDN	digital data network	数字数据网
DDR	double data rate	双倍数据速率
DDR SDRAM	double data rate SDRAM	双倍速率同步动态随机存储器
DE	discard eligibility	适合丢弃
DEC	Digital Equipment Corp.	数字设备公司, DEC 公司(美国)
DEDALUS	deductive algorithm Ur-synthesizer	演绎算法 Ur 综合器
DEDS	discrete event dynamic system	离散事件动态系统
DENDRAL	dendritic algorithm	树枝状算法
DES	data encryption standard	数据加密标准
DES	distributed expert system	分布式专家系统
DF	don't fragment	禁止分段
DFA	deterministic finite automaton	确定型有限自动机
DFD	data flow diagram	数据流程图
DFG	data flow graph	数据流图
DFSA	deterministic finite state automaton	确定型有限[状态]自动机
DFT	design for testability	可测性设计
DFT	discrete Fourier transform	离散傅里叶变换
DI	data input	数据输入
DIPS	Dhrystone instructions per second	条 Dhrystone 指令每秒
DIS	distributed interactive simulation	分布交互式仿真
DIS	draft international standard	国际标准草案
DLAT	directory lookaside table	目录检测表

缩略语	英文全名	中文译名
DLBA	deterministic linear bounded automaton	确定型线性有界自动机
DLCI	data link connection identifier	数据链路连接标识符
DLT	digital linear tape	数字线性磁带(技术)
DM	data mining	数据挖掘
DM	delay modulation	延迟调制(码)
DMA	direct memory access	直接存储器存取
DME	distributed management environment	分布式管理环境
DML	data manipulation language	数据操纵语言
DMRP	distributed manufacturing resource planning	分布式制造资源规划
DMS	data management system	数据管理系统
DNA	deoxyribose nucleic acid	脱氧核糖核酸
DNA	Digital Network Architecture	数字网络体系结构(美国 DEC 公司)
DNC	direct numerical control	直接数值控制,直接数控
DO	data output	数据输出
DOC	diskonchip	片上固态盘
DocBook	document book	电子书
DOCF	data operation control field	[数据]操作控制字段
DOCSIS	data over cable service interface specification	通过电缆服务接口传输数据规范
DOD	Department of Defense	国防部(美国)
DOES	digital optoelectronic switch	数字光电开关
DOK	diskonkey	加密固态盘
DOM	diskonmodule	模板上固态盘
DOM	document object model	文档对象模型
DOR	dimension oriented routing	按维寻径
DOS	Disk Operating System	磁盘操作系统
DPCM	differential pulse code modulation	差分脉码调制
DPDA	deterministic pushdown automaton	确定型下推自动机
DPSK	diffirential phase shift keying	差分相移键控
DQDB	distributed queue dual bus	分布式队列双总线
DRAM	dynamic random access memory	动态随机存储器
DRM	distributed resource management	分布的资源管理
DS	network differential serices	网络区分服务
DSC	decision support center	决策支持中心
DS3	digital signal level 3	第 3 级数字信号(数据传输速率)
DSI	data stream interface	数据流接口(速率)
DSL	data subscriber line	数字用户专用线

缩略语	英文全名	中文译名
DSM	distributed shared memory	分布式共享存储器, 分布式共享主存
DSP	digital signal processing	数字信号处理
DSP	digital signal processor	数字信号处理器
DSS	decision support system	决策支持系统
DSSA	domain specific software architecture	领域特定的软件体系结构
DSSD	data structured system development	数据结构化系统开发
DSSS	direct-sequence spread-spectrum	直接序列扩频
DSV	digital sum variation	“数字和”变化量
DTD	document type definition	文档类型定义
DTE	data terminal equipment	数据终端设备
DTL	diode-transistor logic	二极管-晶体管逻辑电路
DTMF	dual tone multiple frequency	双音多频
DTP	desktop publishing system	桌面出版系统
DUT	device under test	被测器件
DVD	digital versatile disc	数字多用途光盘
DVI	digital video interactive	交互式数字视频, 交互式数字视频 (系统)
DXF	Drawing Graphics Exchange format	绘图交换文件格式
EAN	European article number	欧洲商品(条码), EAN(条码)
EARC	Extraordinary Administrative Radio Conference	无线电特别行政会议
EAROM	electrically alterable PROM	电可修改只读存储器
EB	electronic beam	电子束
EBCDIC	Extended Binary Coded Decimal Interchange Code	扩充的二-十进制交换码, EBCDIC 码
EBL	explanation-based learning	基于解释学习
EC	electronic commerce	电子商务
ECC	error correcting code	纠错码
ECIRC	European Computer Industry Research Center	欧洲计算机工业研究中心(德国)
ECL	emitter coupled logic	射极耦合逻辑[电路]
ECMA	European Computer Manufacturers Association	欧洲计算机制造商协会
ECML	electronic-commerce model language	电子商务模型语言
ECSS	extendible computer system simulator	可扩充的计算机系统模拟器
EDA	electronic design automation	电子设计自动化
EDB	extensional database	外延数据库
EDC/ECC	error detection/error correction code	检纠错码
EDI	electronic data interchange	电子数据交换
EDI II	enterprise data integration	企业数据集成

缩略语	英文全名	中文译名
EDIMS	EDI messaging system	EDI 消息处理系统
EDIN	EDI notification	EDI 通知
EDI-UA	EDI user agent	EDI 用户代理
EDO	extended data output	数据扩展输出
EDSAC	electronic discrete sequential automatic computer	电子离散时序自动计算机,EDSAC 计算机
EDVAC	electronic discrete variable automatic computer	电子离散变量自动计算机,EDVAC 计算机
EEPROM	electrically erasable programmable read only memory	电可擦[可]编程只读存储器
EER	equal error rate	等错误率点
EF	expedited forwarding	加速转发
EFM	eight-fourteen modulation	8/14 调制(码)
EGA	enhanced graphics adapter	增强型彩色图形适配器
EGP	exterior gateway protocol	外部网关协议
EIA	Electronic Industries Association	(美国)电子工业协会
EIDE	enhanced IDE	增强型集成驱动电子线路接口
EIR	equipment identity register	设备识别寄存器
EISA	extended industry standard architecture	扩展工业标准体系结构
EJB	Enterprise JavaBeans	企业 Java 组件
ELI	extensible language 1	可扩充语言 1
EMC	electromagnetic compatibility	电磁兼容[性]
EMI	electromagnetic interference	电磁干扰
EMSP	enhanced modular signal processor	增强型模块化信号处理器
ENIAC	electronic numerical integrator and calculator	电子数字积分器和计算器,ENIAC 计算机
ENRZ1	enhanced non-return-to-zero change on one	增强不归零 1 制
ENRZ	enhanced non-return-to-zero	增强不归零制
EPIC	explicitly parallel instruction computing	显式并行指令计算
EPL	Experimental Programming Language	试验性程序设计语言
EPROM	erasable programmable read only memory	可擦[可]编程只读存储器
EREW	exclusive read exclusive write	互斥读写 X
ERP	enterprise resource planning	企业资源规划
ESCON	Enterprise System Connection Architecture	(IBM 公司)企业系统连接体系结构
ESDI	enhanced small device interface	增强型小设备接口,ESDI 接口
ES	end system	端系统
ES	event scheduling	事件调度(法)
ESP	encapsulation security payload	封装安全负载

缩略语	英文全名	中文译名
ESPRDIT	European Strategic Programme for Research and Development in Information Technology	欧洲信息技术战略研究和发展计划
ET	edge table	边表
ETS	expertise transfer system	专业知识转换系统
ETSI	European Telecommunication Standards Institute	欧洲电信标准协会
EUC	extended UNIX code	扩展的 UNIX 码
EXCA	exchangeable card architecture	可互换板卡体系结构
FACS	facial action coding system	面部动作编码系统
FA	factory automation	工厂自动化
FAR	false acceptance rate	错误接受率
FAT	file allocation table	文件分配表
FC	fiber channel	光纤通道
FC	flip-chip	倒装焊
FC	frame control	帧控制
FCS	fieldbus control system	现场总线控制系统
FCS	frame check sequence	帧检验序列
FD	frequency doubling	倍频(制)
FDD	floppy disk drive	软磁盘驱动器
FDDI	fiber distributed data interface	光纤分布式数据接口
FDMA	frequency division multiple access	频分多址接入
FDM	frequency division multiplexing	频分多路复用
FECN	forward explicit congestion notification	前向显式拥塞通知
FEL	future events list	将来事件表
FEP	front-end processor	前端处理器
FET	field effect transistor	场效应晶体管
FFD	fast flash disk	快闪固态盘
FFM	form flow model	表格流模型
FFS	fast file system	快速文件系统
FFT	fast Fourier transform	快速傅里叶变换
FGSPEC	functional graphical specification	函数图解规约(语言)
FHSS	frequency-hopping spread-spectrum	跳频扩频
FIFO	first in, first out	先进先出
FILO	first in, last out	先进后出
FLAIR	functional language articulated interactive resource	函数式语言接合的交互资源
FM	frequency modulation	调频(制)
FM-AM	file management-access method	文件管理存取方法

缩略语	英文全名	中文译名
FMP	file management protocol	文件管理协议
FMP	flow model processor	流程模型处理机
FMR	false match rate	错误匹配率
FMS	file management system	文件管理系统
FMS	flexible manufacturing system	柔性制造系统
FNMR	false non match rate	错误非匹配率
FORMAC	Formula Manipulation Compiler	公式处理编译程序(语言)
FORTTRAN	Formula Translation	公式翻译(语言)
FP	functional programming	函数式程序设计
FPGA	field programmable gate array	现场可编程门阵列
FPLA	field-programmable logic array	现场可编程逻辑阵列
FPM	fast page mode	快速页面模式
FPM	FTAM protocol machine	FTAM 协议机, 文件传送、存取和管理协议机
FPS	fast packet switching	快速分组交换
FPS	floating-point system	浮点系统
FR	frame relay	帧中继
FRAM	ferroelectron RAM	铁电随机存取存储器
FS	frame state	帧状态
FSA	finite state automaton	有限[状态]自动机
FSK	frequency shift keying	频移键控
FSM	finite state machine	有限状态自动机
FTAM	file transfer, access and management	文件传送、存取和管理
FT-AM	file transfer-access method	文件传送存取方法
FTP	File Transfer Protocol	文件传送协议
FTTC	fiber-to-the-curb	光纤到路边
FUG	functional unification grammar	功能合一语法
GA	gate array	门阵列
GaAsIC	GaAs integrated circuit	砷化镓集成电路
GaAsMOS	GaAs MOS	砷化镓 MOS
GAL	generic array logic	通用阵列逻辑[电路]
GASP	general activity simulation program	通用活动仿真程序
GCD	greatest common divisor	最大公约数
GCP	ground control point	地面控制点
GCR	group coded recording	成组编码记录
GDI	graphics device interface	图形设备接口

缩略语	英文全名	中文译名
GDSS	group decision support system	群体决策支持系统
GEO	geostationary earth orbit	对地静止轨道
GFC	generic flow control	一般流量控制
GGCA	Gemetric Graphics Content Architecture	几何图形内容体系结构
GGF	Global Grid Forum	全球网格论坛
GHC	guarded Horn clause	保护性霍恩子句
GII	global information infrastructure	全球信息基础设施
GIIT	graphical input interactive technique	图形输入交互技术
GIS	geographic information system	地理信息系统
GKS-3D	graphical kernel system for three dimensions	三维图形核心系统
GKS	graphical kernel system	图形核心系统
GKSM	GKS metafile	GKS 元文件(接口);GKS 图形标准
4GL	fourth generation language	第四代语言
GLSI	gigantic large scale integration	巨大规模集成[电路]
GMD	Gesellschaft fur Mathematik und Datenverbetung	德国政府科学研究中心
GMM	glued multi-resolution model	黏合多分辨率模型
GOOD	general object-oriented software development	通用的面向对象软件开发
GOSIP	government open systems interconnection profile	政府开放系统互连功能轮廓
GP	graphic processor	图形处理器
GPIB	general purpose interface bus	通用接口总线
GPIO	general purpose input/output	通用输入输出(接口)
GPS	general problem solving system	通用问题求解系统
GPS	general problem solving	一般问题求解
GPS	global position system	全球定位系统
GPSSG	generalized phrase structure grammar	广义短语结构语法
GPSS	general purpose systems simulation	通用系统模拟(语言),GPSS(语言)
GPSS	general purpose systems simulator	通用系统仿真器
GRASS	geographical resources analysis support system	地理资源分析支持系统
GRIP	graphics interactive programming	图形交互程序设计
GSM	Global System for Mobile Communication	全球移动通信系统
GSS	guided self-scheduling	引导[自]调度
GT	group technology	群[成组]技术
GUI	graphical user interface	图形用户界面
GUIDS	graphical user interface description system	图形用户界面描述系统
GUSTO	Globus Uniquitous Supercomputing Testbed Organization	Globus 随遇超级计算试验床组织
GWUIMS	George Washington User Interface Management System	乔治·华盛顿用户界面管理系统

缩略语	英文全名	中文译名
HAL	hardware abstraction layer	硬件抽象层
HAMT	human-aided machine translation	人助机译
HBT	heterojunction bipolar transistor	异质结双极[型]晶体管
HCI	human computer interaction	人机交互(接口)
HCSS	high-level computer system simulator	高级计算机系统模拟器
HDA	head disk assembly	头盘组合件
HDAM	hierarchical direct access method	分层直接存取法
HDD	hard disk drive	硬磁盘驱动器
HDL	hardware description language	硬件描述语言
HDLC	high-level data link control (procedures)	高级数据链路控制(规程)
HDSL	high-bit-rate digital subscriber line	高比特率数字用户专用线
HDSS	holographic data storage system	全息数据存储系统
HEC	header error control	信头差错控制
HEMT	high electron mobility transistor	高电子迁移率晶体管
HFC	hybrid fiber coaxial cable	混合光纤同轴电缆
HFET	heterojunction field effect transistor	异质结场效应晶体管
HIDAM	hierarchical indexed direct access method	分层索引直接存取法
HIPO	hierarchical input, processing, output	层次输入-处理-输出(图)
HIPPI	high performance parallel interface	高性能并行接口
HISAM	hierarchical indexed sequential access method	分层索引顺序存取法
HLR	home location register	归属位置寄存器
HLS	hue-lightness-saturation	色调-亮度-饱和度(模型)
HMC	human-computer interaction and communication	人机交互和通信
HMD	head-mounted display	头盔显示器
HMM	hidden Markov model	隐式马尔可夫模型
HOLAP	hybrid online analysis processing	混合联机分析处理
HOOD	hierarchical object-oriented design	层次的面向对象设计
HP	Hewlett-Packard Co.	惠普公司, HP 公司
HPF	high pass filter	高通滤波器
HPF	high performance FORTRAN	高性能 FORTRAN(语言)
HPIB	Hewlett-Packard interface bus	惠普[公司]接口总线
HPSC	head-driven phrase structure grammar	中心词驱动短语文法
HQS	high quality screening	高质量挂网(技术)
HSAM	hierarchical sequential access method	分层顺序存取法
HSV	hue-saturation-value	色调-饱和度-明度值(模型)
HTML	hyper text markup language	超文本置标语言

缩略语	英文全名	中文译名
HTTP	hypertext transfer protocol	超文本传送协议
HyTime	Hypermedia Time-based Structuring Language	基于时间的超媒体结构化语言
IBE	ion beam etching	离子铣
IBG	interblock gap	[记录]数据块间隙
IBM	International Business Machines Corp.	国际商业机器公司, IBM 公司(美国)
IBM	image-based modeling	基于图像的造型
IBM-DOS	IBM Disk Operating System	IBM 公司磁盘操作系统
IBR	image-based rendering	基于图像的绘制
IC	integrated circuit	集成电路
ICA	independent component analysis	独立元分析
ICAI	intelligent computer-assisted instruction	智能计算机辅助教学
ICAM	integrated computer-aided manufacturing	综合计算机辅助制造
ICC	International Color Consortium	国际颜色联盟
ICCC	International Computer Communications Conference	国际计算机通信会议
ICMP	internet control message protocol	互连网控制消息协议
ICMS	integrated circuit message switch	集成电路信息交换设备
ICN	interconnection network	互联网
ICN	information control network	信息控制网络(模型)
ICOT	Institute for New Generation Computer Technology	新一代计算机技术研究所(日本)
ICP	ion coupling	感应耦合等离子
ID	icon dictionary	[基本]图符字典
IDA	The Institute for Defense Analysis	(美国)国防部防御分析研究所
IDB	intensional database	内涵数据库
IDE	integrated data environment	数据集成环境
IDE	integrated drive electronics	集成驱动器电子线路(接口), IDE (接口)
IDE	interactive data entry	交互式数据录入
IDE	Interactive Development Environments	交互式开发环境(公司)
IDEA	international data encryption algorithm	国际数据加密算法
IDEF	integrated computer-aided manufacturing definition method	综合计算机辅助制造工程定义方法, ICAM 定义方法
IDFT	inverse discrete Fourier transform	离散傅里叶逆变换
IDL	interface definition language	接口定义语言
IDLC	integrated digital loop carrier	集成数字环载波
IDMS	integrated data base management system	综合数据库管理系统
IDN	integrated digital network	综合数字网

缩略语	英文全名	中文译名
IDS	information data system	信息数据系统
IDSS	intelligent decision support system	智能决策支持系统
IDT	interactive design tool	交互设计工具
IE	ion etching	等离子蚀刻
IEC	International Electrotechnical Commision	国际电工委员会
IEEE	Institute of Electrical and Electronics Engineers	电气和电子工程师学会(美国)
IETF	Internet Engineering Task Force	因特网工程任务部
IETM	interactive electronic technical manual	交互式电子技术手册
IFIP	International Federation for Information Processing	国际信息处理联合会
IFS	installable file system	可安装文件系统
IGBT	insulated gate bipolar transistor	绝缘栅双极晶体管
IGES	initial graphical exchange specification	初始图形交换规范
IGP	interior gateway protocol	内部网关协议
IGRP	interior gateway router protocol	内部网关路由协议
IGS	interactive graphics system	交互图形系统
IHL	Internet header length	因特网报头长度
II	initial interval	启动间距
IIL	integrated injection logic	集成注入逻辑电路
IKE	Internet key exchange	Internet 密钥交换
I ² L	integrated injection logic	集成注入逻辑[电路]
ILP	instruction level parallelism	指令级并行性
ILPP	instruction level parallel processing	指令级并行处理
IMP	interface message processor	接口[报文处理]机
IMS	information management system	信息管理系统
IN	intelligent network	智能网
INGRES	interactive graphic and retrieval system	交互式图示检索系统
INTERLISP	interactive LISP	交互 LISP 语言
IOD	input/output device	输入输出设备
I/O	input/output	输入输出
IP	internet protocol	网际协议
IPC	instructions per cycle	每个周期平均执行的指令数
IPD-CMM	capability maturity model for integrated product development	集成产品开发能力成熟度模型
IPG	information power grid	信息资源网格
IPI	intelligent peripherals interface	智能外围设备接口, IPI 接口
IPL	information processing language	信息处理语言

缩略语	英文全名	中文译名
IPSEC	IP security	(IETF 制定的)网际协议安全性
IPv6	Internet Protocol version 6.0	第 6.0 版网际协议
Ipv4	Internet Protocol version 4.0	第 4.0 版网际协议
IPX	internetwork packet exchange	网间数据包交换(协议)、网际包交换(协议)
IQLA	International Quantitative Linguistics Association	国际计量语言学学会
IR	information retrieval	信息检索
IR	infrared radiation	红外
IR	instruction register	指令寄存器
μIR	microinstruction register	微指令寄存器
IRS	information retrieval system	信息检索系统
IS	intermediate system	中间系统
IS	international standard	国际标准
ISA	industry standard architecture	工业标准体系结构
ISAM	indexed sequential access method	索引顺序存取方法
iSCSI	Internet SCSI	因特网 SCSI 接口
ISDN	integrated services digital network	综合业务数字网
ISDOS	information system design and optimization system	信息系统设计和优化系统
ISG	inter-sector gap	扇区后间隙
IS-IS	intermediate system to intermediate system	IS-IS(协议)
ISO	International Standards Organization	国际标准化组织
ISO/OSI	International Organization for Standardization Open System Interconnection Model	ISO/OSI 模型、国际标准化组织-开放系统互连基准模型
ISP	Internet service provider	因特网服务提供者
ITSEC	Information Technology Security Evaluation Criteria	信息技术安全评估准则
ITT	International Telephone and Telegraph Corporation	国际电话电报公司(美国)
ITU	International Telecommunication Union	国际电信联盟
IVD	integrated voice and data	综合语音与数据
JCL	job control language	作业控制语言
JDBC	Java database connectivity	Java 数据库互连
JIS	Japanese Industrial Standard	日本工业标准
JIT	just in time	及时
JITP	just in time production	及时生产
JLD	joint-dependent local deformation	关节相关的局部变形
JOSS	Johnniac open-shop system	Johnniac 开放系统(语言)
JOVIAL	Jules own version of the international algorithmic language	国际算法语言的朱尔斯专用文本

缩略语	英文全名	中文译名
JPEG	Joint Photographic Expert Group	联合静止图像专家组
JPL	Jet Propulsion Laboratory	喷气推进实验室
JSD	Jackson system development	Jackson 系统开发[方法]
JSP	Jackson structured programming	Jackson 结构化程序设计
JTC	Joint Technical Committee	联合技术委员会
JTM	job transfer and manipulation	作业传送和操纵
JVM	Java virtual machine	Java 虚拟机
KADS	knowledge-aided document system	知识辅助文档系统
KAS	knowledge acquisition system	知识获取系统
KB	knowledge base	知识库
KBMS	knowledge base management system	知识库管理系统
KBS	knowledge-base system	知识库系统
KDC	key distribution center	密钥分发中心
KDD	knowledge discovery in database	数据库知识发现
KL	kernel language	核心语言
KQML	knowledge query and manipulation	知识查询与操纵语言
KRL	knowledge representation language	知识表示语言
KSR	keyboard send/receive	键盘发送接收
KWIC	keyword in context indexing	上下文中的关键字检索
KWIPS	kilo Whetstone instructions per second	千条 Whetstone 指令每秒
LAN	local area network	局域网
LAP	link access procedure	链路接入规程
LAP B	link access procedure-balanced	平衡型链路接入规程
LAP D	link access control procedure on the D-channel	D 通道链路接入规程
LAP F	Link Access Procedure for Frame-mode hearer services	用于帧中继的链路访问规程
LAT	local area transport	局部区域运输(协议)
LBA	linear bounded automaton	线性有界自动机
LBA	logic block addressing	逻辑块编址(模式)
LCD	liquid crystal display	液晶显示,液晶显示器
LCF	logic for computable functions	可算函数逻辑
LCF	low cost fiber	低成本光纤
LCP	link control protocol	链路控制协议
LCP	logical construction of programs	程序[的]逻辑构造,程序的逻辑结构(方法)
LCS	logical construction of systems	系统[的]逻辑构造
LDAP	light weight directory access protocol	轻量级目录服务协议

缩略语	英文全名	中文译名
LDD	lightly doped drain	轻掺杂漏区
LDL	logical data base level	逻辑数据库级
LED	light-emitting diode	发光二极管
LEO	low earth orbit	近地轨道
LFA	local feature analysis	局部特征分析
LFG	lexical functional grammar	词汇功能语法
LFSR	linear feedback serial register	线性反馈串行寄存器
LFU	least frequently used	最不常用
LGMR	laser guidance magnetic recording	激光制导磁记录
LinMOS	linear CMOS	线性 CMOS
LIP	loop initialize process	环初始化进程
LISP	List Processing	表处理语言, LISP 语言
LIW	long instruction word	长指令字
LLC	logical link control	逻辑链路控制
LMDS	local multipoint distribution services	本地多点分配业务
LMSC	LAN/MAN Standards Committee	局域网和城域网标准委员会
LOS	logical output structure	逻辑输出结构
LP	low pass	低通(滤波器)
LPC	linear prediction coding	线性预测编码
LPC	linear predictive coefficient	线性预测系数
LPC	local procedure call	本地过程调用
LPCVD	low pressure chemical vapor deposition	低压化学气相淀积
LPF	low pass filter	低通滤波器
LPS	logical processing structure	逻辑处理结构
LPSL	low press field effect transistor	低夹断场效应晶体管
LR	location register	定位寄存器
LRC	longitudinal redundancy check	水平冗余检验, 纵向冗余检验
LRU	least recently used	最近最少使用
LSAP	link services access point	链路服务访问点
LSB	least significant bit	最低有效位
LSI	large scale integration	大规模集成[电路]
LSP	link state packet	链路状态分组
LSSD	level sensitive scan design	电平敏感扫描设计
LTO	linear tape open	线性磁带(系统)开放(技术)
LU	line unit	线路单元
LUT	look-up table	查找表

缩略语	英文全名	中文译名
LVD	laser video disc	激光视盘
LWIRD	long-wave length infrared detector	长波长红外探测器
MAC	medium access control	介质访问控制
MAHT	machine-aided human translation	机助人译
MAI	multiple access interference	多址接入干扰
MANDATE	multiline automatic network diagnostic and transmission equipment	多线路自动网络诊断和传输设备
MAN	metropolitan area network	城域网
MAP	manufacturing automation protocol	制造自动化协议
MAP	mobile application part	移动应用部分
MAR	memory address register	存储器地址寄存器
MARGIE	meaning analysis, response generation and inference on English	英语的语义分析、反应生成和推断(系统)
MathML	Mathematical Markup Language	数学置标语言
MAU	medium access unit	介质访问单元
MAVICA	magnetic video camera	磁性视频摄像
MB	model base	模型库
MBE	molecular beam epitaxy	分子束外延
MBMS	model base management system	模型库管理系统
MBR	memory buffer register	存储器缓冲寄存器
MCA	micro channel architecture	微通道体系结构
MCC	Micro-electronics and Computer Technology Corp.	微电子与计算机技术公司(美国)
MCC	Multiple Computer Complex	多计算机联合体(美国)
MCI	Microwave Communications, Inc.	微波通信公司(美国)
MCM	multichip module	多芯片模块
MCPC	multi-channel per carrier	单载波多路
MCS	microcomputer system	微型计算机系统
MCU	machine control unit	机器控制单元
MCU	microcontrol unit	微控制器
MCU	micro-programmed control unit	微程序控制器
MDA	model-driven architecture	模型驱动的体系结构
MDA	monochrome display adapter	单色显示适配器
MDI	medium dependent interface	介质相关接口
MEO	middle earth orbit	中间地球轨道
MESFET	metal-semiconductor field effect transistor	金属-半导体场效应晶体管
MF	more fragment	尚有分段

缩略语	英文全名	中文译名
MFLOPS	million floating point operations per second	百万[次]浮点运算每秒
MFM	modified frequency modulation	改进调频(制)
MHEG	Multimedia and Hypermedia Expert Group	多媒体和超媒体专家组
MHS	message handling system	消息处理系统
MIB	management information base	管理信息库
MIC	memory in cassette	磁带盒内存储器(技术)
MIDAS	modified integration digital to analog simulator	改进的集成数模仿真器
MIDI	musical instrument digital interface	乐器数字接口
MIG	metal-in-gap	隙含金属(磁头)
MIMD	multiple-instruction stream multiple-data stream	多指令流多数据流
MIME	multipurpose Internet mail extension	多用途因特网电子邮件扩展(协议)
MIPS	million instructions per second	百万[条]指令每秒
MISD	multiple-instruction stream single-data stream	多指令流单数据流
MIS	management information services	管理信息服务
MIS	manufacturing information system	制造信息系统
MIT	management information tree	管理信息树
MIT	Massachussetts Institute of Technology	麻省理工学院(美国)
ML	macrolanguage	宏加工语言
ML	metalanguage	元语言
MM	main memory	主存储器
MMCD	multimedia CD	多媒体数字光盘
MMDS	multichannel multipoint distribution services	多路多点分配业务
MMIC	monolithic microwave integrated circuit	单片微波集成电路
MMS	manufacturing message specification	制造报文规范
MMU	main memory unit	主存储器
MMU	memory management unit	存储器管理部件, 存储管理部件
MO	managed object	被管对象
MOCVD	metal organism CVD	金属有机物化学汽相淀积法
MOD	moving object database	移动对象数据库
MODFET	modulation doped field effect transistor	调制掺杂场效应晶体管
MOLAP	multidimensional online analysis processing	多维联机分析处理
MOPS	micromonitor operating system	微型监控器操作系统
MOS	metal-oxide-semiconductor	金属-氧化物-半导体
MOSFET	metal-oxide-semiconductor field effect transistor	金属-氧化物-半导体场效应晶体管
MOTIS	message oriented text interchange system	面向消息的正文交换系统

缩略语	英文全名	中文译名
MP	multimedia processor	多媒体处理器
MP	multiprocessor	多处理机
MPC	multimedia personal computer	多媒体个人计算机
MPEG	Moving Picture Expert Group	运动图像专家组
MPI	message passing interface	消息传递接口
MPP	massively parallel processing	大规模并行处理
MQ	message queuing	报文队列
MR	magnetic resistive	磁变阻(磁头)
MRAM	magnetic-resistive RAM	磁电阻随机存取存储器
MRD	machine readable dictionary	机器可读词典
MRP	manufacturing resource planning	制造资源规划
MRP	material requirement planning	物料需求计划
MRU	maximum receive unit	最大接收单元
MS	message store	消息存储[单元]
MS	mobile station	移动台
MSB	most significant bit	最高有效位
MSC	moving switcher	移动交换机
MS-DOS	Microsoft Disk Operating System	微软公司磁盘操作系统
MSE	mean square error	均方误差
MSI	medium scale integration	中规模集成[电路]
MSIPS	million synchronous instructions per second	百万条同步指令每秒
MSN	Manhattan street network	曼哈顿街道网
MSS	MAN switching system	城域网交换系统
MSS	mass storage system	海量存储系统
MSS	multispectral scanner	多光谱扫描仪
MT	machine translation	机器翻译
MTA	message transfer agent	消息传送代理
MTBF	mean time between failures	平均故障间隔时间
MTL	message transfer sublayer	消息传送子层
MTOPS	millions of theoretical operations per second	百万次理论运算每秒
MTS	message transfer system	消息传送系统
MTTF	mean time to failure	平均无故障时间
MTTR	mean time to repair	平均修复时间
MTU	maximum transfer unit	(网络)最大传输单元
MTX	mobile telephone exchange	移动电话交换局

缩略语	英文全名	中文译名
MVL	multivalued logic	多值逻辑电路
MVS	multiple virtual storage (operating system)	多虚拟存储器(操作系统)
MWIPS	million Whetstone instructions per second	百万条 Whetstone 指令每秒
MWM	MOTIF window manage	MOTIF 窗口管理器
NAP	network access process	网络存取进程
NAS	network attached storage	附网存储
NAS	National Academy of Sciences	(美国)国家科学院
NASA	National Aeronautics and Space Administration	(美国)国家宇航局
NAT	network address translation	网络地址翻译
NATO	North Atlantic Treaty Organization	北大西洋公约组织
NAV	network assigned vector	网络配给向量
NBM	narrow band modulation	窄带调制(码)
NBS	National Bureau of Standards	国家标准局(美国)
NC	network computer	网络计算机
NC	numerical control	数值控制,数控
NCC	network control center	网络控制中心
NCP	network control protocol	网络控制协议
NCSA	National Computational Science Alliance	(美国)国家计算科学同盟会
NCSC	National Computer Security Center	(美国)国家计算机安全中心
NDBMS	network data base management system	网络数据库管理系统
NDC	normalized device coordinate system	规格化设备坐标系
NDD	network data dictionary	网络数据字典
NDS	domain name system	域名系统
NE	network element	网络元素
NEBULA	natural electronic business users language	电子商业用户自然语言
NEC	Nippon Electric Co.	日本电气公司, NEC 公司
NELIAC	Navy Electronics Laboratory International ALGOL Compiler	海军电子实验室国际 ALGOL 编译程序
NFA	nondeterministic finite automaton	非确定型有限自动机
NFS	Network File System	网络文件系统(美国 Sun 公司)
1NF	the first normal form	第一范式
2NF	the second normal form	第二范式
3NF	the third normal form	第三范式
4NF	the fourth normal form	第四范式
NGI	next generation Internet	下一代因特网
NIC	network information center	网络信息中心

缩略语	英文全名	中文译名
N-ISDN	narrowband integrated services digital network	窄带综合业务数字网
NIST	National Institute of Standards and Technology	国家标准化技术研究所(美国)
NLN	neural logic network	神经逻辑网络
NLP	natural language processing	自然语言处理
NLSP	Netware Link Service Protocol	Netware 链路服务协议
NLU	natural language understanding	自然语言理解
NMC	network management center	网[络]管[理]中心
NMOS	N-channel metal-oxide-semiconductor	N 沟道金属-氧化物-半导体
NMR	nuclear magnetic resonance	核磁共振
NMS	network management station	网络管理站
NMT	network management terminal	网络管理终端
NNI	network-network interface	网间接口
Non-GEO	non-geostationary earth orbit	对地静止轨道
NOS	network operating system	网络操作系统
NOW	network of workstation	工作站网
NP	noun phrase	名词短语
NPACI	National Partnership for Advanced Computational Infrastructure	国家高级计算设施联合会(美国)
NPL	nonprocedural language	非过程语言
NPR	non-photorealistic rendering	非真实感图形绘制
NPT	non-packet terminal	非分组式终端,非包式终端
NRC	National Research Council	国家科学研究委员会(加拿大)
NRM	normal response mode	正常响应模式
NRZ1	non-return-to-zero change on one	逢 1 变化不归零制(码)
NRZ	non-return-to-zero	不归零制(记录格式)
NSA	National Security Agency	国家安全局(美国)
NSF	National Science Foundation	国家科学基金会(美国)
NSS	network and switching subsystem	网络和交换系统
NURBS	non-uniform rational B-spline	非均匀有理 B 样条
NV RAM	non-volatile RAM	非易失性随机存储器
OA	office automation	办公自动化
OAS	open application system	开放应用系统
OBE	office by example	实例办公语言
OC-48	optical carrier, level 48	光载波信号线路 48 级
OC-12	optical carrier, level 12	光载波信号线路 12 级
OC-3	optical carrier, level 3	光载波信号线路 3 级

缩略语	英文全名	中文译名
OCR	optical character reader	光学字符阅读机
OCR	optical character recognition	光学字符识别
ODBC	open database connection	开放式数据库连接
ODBC	open database connectivity	开放式数据库互连
ODP	open distributed processing	开放式分布处理
OEIC	opto-electronic integrated circuit	光电集成电路
OEM	object exchange model	对象交换模型
OEM	original equipment manufacturer	初始设备制造厂
OGSA	open grid services architecture	开放式网格服务体系结构
OIS	office information system	办公信息系统
OLAP	online analysis processing	联机分析处理
OLE	object linking and embedding	对象链接与嵌入(功能)
OLTP	on-line transaction processing	联机事务处理
OLTP	on-line transaction processing system	联机事务处理系统
OMC-R	operation and maintenance center-radio	操作与维护中心-无线电
OMC-S	operation and maintenance center-switching	操作与维护中心-交换机
OMG	Object Management Group	对象管理组织
OMG-IDL	OMG Interface Definition Language	OMG 组织提出的接口定义语言
OMR	optical mark reader	光学标记阅读机
OMT	object modeling technique	对象模型化技术
ONU	optical network unit	光纤网部件
OO	object-oriented development method	OO 方法,面向对象的开发方法
OOA	object-oriented analysis	面向对象的分析
OODBMS	object-oriented database management system	面向对象数据库管理系统
OOD	object-oriented design	面向对象的设计
OOI	object-oriented implementation	面向对象的实现
OOP	object-oriented programming	面向对象的程序设计
OOSD	object-oriented structured design	面向对象的结构化设计
OPS	operations per second	运算(次数)每秒
OR	operation ratio	运转率,运行率
ORB	object request broker	对象请求代理
ORC	optical rectangular code	最佳距阵码
ORDBMS	object-relational database management system	对象-关系数据库管理系统
OS	open system	开放系统
OS	operation system	运营系统
OSF	Open Software Foundation	开放软件基金会

缩略语	英文全名	中文译名
OSI	open system interconnection	开放系统互连
OSIE	open system interconnection environment	开放系统互连环境
OSI/RM	open system interconnection reference model	开放系统互连基准(参考)模型
OSPF	open shortest path first	开放式最短通路优先(算法)
Oss	optical carriers	光纤载体
OTM	office task management	办公任务管理(模型)
OTP EPROM	one time programmable EPROM	一次可编程 EPROM
OUM	ovonic unified memory	相变存储器
PABX	private automatic branch exchange	专用自动小交换机
PAC	probably approximately correct	概率近似正确(学习模型)
PAD	packet assembler/disassembler	分组组装拆器,包装拆器
PAD	problem analysis diagram	问题分析图
PAL	programmable array logic	可编程阵列逻辑[电路]
PAM	pulse amplitude modulation	脉[冲]幅[度]调制
PAP	password authentication protocol	口令鉴别协议
PARC	Palo Alto Research Center	Palo Alto 研究中心(美国 Xerox 公司)
PARLOG	Parallel Programming In Logic	并行逻辑程序设计(语言), PAR-LOG(语言)
PASS	program alternative simulation system	程序交替模拟系统
PAN	personal area network	私人网
PBX	private branch exchange	专用小交换机
PC	personal computer	个人计算机,PC 机
PC	programmable controller	可编程控制器
PC	program counter	程序计数器
PCA	principal component analysis	主成分分析
PCB	printed circuit board	印制[电路]板
PC-DOS	Personal Computer-Disk Operating System	个人计算机磁盘操作系统
PCF FORTRAN	parallel context-free FORTRAN	并行上下文无关 FORTRAN(语言)
PCFG	probabilistic context-free grammar	概率型上下文无关文法
PCI	peripheral component interconnection	外围部件互连
PCI	protocol control information	协议控制信息
PCI, PCI-X	Peripheral Connection Interface	外围部件互连(接口)
PCL	printer control language	打印机控制语言
PCMCIA	Personal Computer Memory Card International Association	个人计算机存储卡国际协会
μPC	microprogram counter	微程序计数器
PCM	pulse code modulation	脉[冲编]码调制

缩略语	英文全名	中文译名
PCN	personal communication network	个人通信网
PCS	Personal Conferencing Specification	个人会议标准(规范)
PCT	personal construct theory	个人构造理论
PCTE	portable common tool environment	可移植的公共工具环境
PCWG	Personal Conferencing Work Group	个人会议工作组
PD	phase detector	鉴相器
PDA	personal digital assistant	个人数字助理
PDA	pushdown automaton	下推自动机
PDAU	physical delivery access unit	物理投递访问单元
PDDI	fiber distributed data interface	光纤分布式接口
PDES	product data exchange specification	产品数据交换标准
PDF	portable document format	可移植表示格式(语言)
PDL	program description language	程序描述语言
PDL	program design language	设计[性]程序语言
PDM	product data management	产品数据管理
PDN	public data network	公用数据网
PDP	parallel distributed processing	并行分布式处理
PDR	processing data rate	数据处理速率
PDS	premises distribution system	宅院布线系统
PDU	packet data unit	分组数据单元
PDU	protocol data unit	协议数据单元
People-CMM	Capability Maturity Model for People	人员能力成熟度模型
PE	phase encoding	调相制,相位编码
PE	plasma etching	等离子蚀刻
PE	processing element	处理部件
PERT	program evaluation and review technique	计划评价与评审技术
PGP	Pretty Good Privacy	良好隐私(事实上的全球电子邮件加密标准)
PHB	per-hop-behavior	每跳行为
PHEMT	psedo-morphic high electro mobility transistor	伪晶高电子迁移率晶体管
PHIGS	programmer's hierachical interactive graphics system	PHIGS 图形标准
PHIGS	programmer's hierarchical interactive graphics standard	程序员分层交互图形标准
PHY	physical sublayer	物理子层
PI	process interactive	进程交互(法)
PIC	programmable intelligent computer	可编程智能计算机
PICS	production information and control system	生产信息与管理系统

缩略语	英文全名	中文译名
PICS	protocol implementation conformance statement	协议实现一致性声明
PID	process identifier	进程标识符
PID	proportional-integral-differential	比例积分微分
PID	proportional-integral-differential controller	比例积分微分控制器, PID 控制器
PIM	parallel inference machine	并行推理机
PIM	processor-in-memory	处理器在内存
PIMOS	parallel inference machine operating system	并行推理机操作系统
PIN	personal identification number	个人认证号
PIO	programmed input/output	程序控制输入输出
PKI	public key infrastructure	公开密钥设施
PLA	programmable logic array	可编程逻辑阵列
PLATO	programmed logic for automatic teaching operations	自动教学业务用程序逻辑
PLATO	programming language for automatic teaching operations	自动教学业务用的程序设计语言
PLD	programmable logic device	可编程逻辑器件
PLL	phase locked logic	锁相逻辑[电路]
PLL	phase-locked loop	锁相环
PLM	programmable logic machine	可编程序逻辑机
PL /1	Programming Language /1	程序设计语言 1, PL/1 语言
PLS	physical signaling	物理信令(号)
PLTL	propositional linear temporal logic	命题线性时态逻辑
PM	phase modulation	调相(制)
PM	physical medidum	物理介质
PM	physical medium	(ATM) 物理介质(子层)
PMA	physical medium attachment	物理介质连接
3PM	3 phase modulation	3 单元调相制
3PM	3 position modulation	3 单元调制(码)
PM	progressive meshes	递进网格
PMD	physical medium-independent sublayer of local area network	局域网物理介质无关子层
PMOS	P-channel MOS	P 沟道 MOS
PNNI	private network to network interface	专用网络间接口
PNN	probabilistic neural network	概率神经网络
PON	passive optical network	无源光纤网
POS	personal operating space	私人工作空间
POS	point of sale terminal	销售点终端
POSIX	portable operating system interface for computer environments	计算机环境的可移植操作系统接口

缩略语	英文全名	中文译名
POSIX	Portable Operating System Interface(UNIX-like)	UNIX 可移植操作系统界面(标准)
POSIX	portable operating system UNIX	可移植的 UNIX 操作系统
POSLA	pitch synchronous overlap add	基音同步叠加
PP	preposition phrase	介词短语
P2P	person to person	个人到个人
PPA	probabilistic pushdown automaton	概率下推自动机
PPP	point-to-point protocol	点对点连接协议
PRAM	parallel random access machine	并行随机存取机
PRML	partial response maximum likelihood	部分响应最大似然(信号处理技术)
PROLOG	Programming in Logic	逻辑程序设计(语言), PROLOG (语言)
P2ROM	product-PROM	产品-可编程只读存储器
PROM	programmable read-only memory	可编程只读存储器
PRPS	phase rectifying power supply	多相整流电源
PSA	problem statement analyzer	问题陈述分析程序
PSDN	packet switched data network	分组交换数据网
PSG	phrase structure grammar	短语结构语法
PSI	personal sequential inference machine	个人顺序推理机
PSK	phase shift keying	相移键控
PSL	problem statement language	问题陈述语言
PSM	problem solving description mechanism	问题求解描述机制
PSOLA	pitch-synchronous overlap add	音调同步重叠相加法
PSOS	plug-in silicon operating system	插入式硅操作系统
PSP	personal software process	个人软件过程
PSTN	public switched telephone network	公用交换电话网
PTE	page table entry	页表项
P to P	person to person	个人到个人
PT	payload type	净荷类型
PUP	PARC universal packet	Palo Alto 研究中心通用分组
PVC	permanent virtual circuit	永久性虚电路
PVM	parallel virtual machine	并行虚拟机
PVP	parallel vectors processors	并行向量处理(机)
QAM	quadratic amplitude modulation	正交调幅
QBE	Query By Example	实例查询语言(美国 IBM 公司)
Qb	quantum bit	量子位
Qbit	quantum bit	量子位

缩略语	英文全名	中文译名
QCIF	quarter-CIF	四分之一 CIF 图像格式
QIC	quarter inch tape	1/4 英寸盒带
QoS	quality of service	服务质量
QR	quick response	快速响应
QUEL	query language	查询语言
RA	ripple adder	行波加法器
RAID	redundant array of inexpensive disk	廉价冗余磁盘阵列
RAM	random access machine	随机存取机
RAM	random access memory	随机存取存储器
RAS	reliability, availability and serviceability	可靠性、可用性及可维[修]性
RB	route bit	路由比特
R/C	return/count	返回-计数(寄存器)
RDA	remote database access	远程数据库访问
RDBMS	relational database management system	关系数据库管理系统
RDRAM	Rambus dynamic random access memory	Rambus 动态随机存储器
RF	radio frequency	射频
RFC	Request for Comments	征求意见文件
RFC2210	Request for Comments 2210	第 2210 号征求意见文件
RG	regular grammar	正则文法, 3 型文法
RGB	red-green-blue	红-绿-蓝(模型)
RIE	reactive ion etching	反应离子蚀刻
RIG	Reuse Library Interoperability Group	复用库互操作组织
RIP	raster image processor	光栅图像处理器, 栅格图像处理器
RIP	routing information protocol	路由信息协议
RISC	reduced instruction set computer	精简指令集计算机
RLLC	run-length-limited code	游程长度受限码
rlogin	remote login	远程登录
RML	requirements modeling language	需求模型化语言
RMON MIB	remote network monitoring management information base	远程网络监控管理信息库
R-M	reed muller	舌簧混沙机
RMON	remote network monitoring	远程网络监控
ROC	receiver operating curve	ROC 曲线
ROLAP	relational online analysis processing	关系联机分析处理
ROM	read-only memory	只读存储器
ROSE	remote operations service element	远程操作服务要素
RPC	remote procedure call	远程过程调用

缩略语	英文全名	中文译名
RPG	report program generator	报表程序生成程序
RPLA	reprogrammable logic array	可重编程逻辑阵列
R-S	Reed-Solomon (code)	里德-索罗门(码)
RS	remote sensing	遥感
RSA	Rivest-Shamir-Adleman (method)	RSA 方法(一种数据安全编码方法)
RSEEXEC	resource-sharing executive	资源共享执行程序
RSL	requirements statement language	需求陈述语言
RS-PC	reed-solomon product code	里德-索罗门产品编码
RSVP	network resource reservation protocol	网络资源预约协议
RTA	rapid thermal anneal	快速退火(技术)
RTC	real time clock	实时时钟
RTD	resonant tunneling device	共振隧穿器件
RTL	register transfer language	寄存器传送语言
RTL	resistor transistor logic	电阻-晶体管逻辑
RTP	Real-time Transfer Protocol	实时传输协议
RTS/CTS	request to send/clear to send	请求发送与清除发送
Rt-VBR	real-time variable bit rate	实时可变比特率(服务)
RWC	real world computing	真实世界计算
SA	structured analysis	结构化分析
SA	source address	源地址
SA	security association	安全关联
SAA	system application architecture	系统应用体系结构
SA-CMM	capability maturity model for software acquisition	软件获取能力成熟度模型
SADT	structured analysis and design technique	结构化分析与设计技术
SAG	SQL Access Group	SQL 访问组
SAGE	semi-automatic ground environment computer	半自动地面防空警戒计算机,赛其 防空警戒计算机
SALT	symbolic algebraic language translator	符号代数语言翻译程序
SAMM	systematic activity modeling method	系统化活动建模方法
SAM	sequential access memory	串行存取存储器
SAM	serial access memory	串行存取存储器
SAN	storage area network	存储区域网
SAP	service access point	服务访问点
SAR	segmentation and reassembly	分段与重组
SAS	serial attached SCSI	串行嵌入式 SCSI 接口
SASE	specific application service element	特定应用服务要素

缩略语	英文全名	中文译名
SATA	serial ATA	串行 ATA 接口
SAX	simple API for XML	XML 的简单应用编程接口
SBI	serial bus interface	串行总线接口
SC	structure chart	结构图
SC	Subcommittee	分委员会(国际标准化组织)
SCCS	source code control system	源[代]码控制系统
SCFL	source coupled FET logic	源耦合场效应晶体管逻辑电路
SCOOP	system for computerization of office processes	计算机化办公事务处理系统
SCR	silicon controlled rectifier	可控硅整流器
SCS	structured cabling system	结构化布线系统
SCSI	small computer system interface	小计算机系统接口, SCSI 接口
SD	structured design	结构化设计
SD	super high density digital video disc	超高密度数字视盘
SDD	system for distributed databases	分布式数据库系统
SDE	shared data environment	数据共享环境
SDE	software development environment	软件开发环境
SDFL	Schottky diode FET logic	肖特基二极管场效应晶体管逻辑电路
SDF-4	stroke density features in four direction	四方向笔画密度特征
SDH	synchronous digital hierarchy	同步数字体系
SDL	structural descriptive language	结构描述语言
SDLC	synchronous data link control	同步数据链路控制(规程)
SDLT	super digital linear tape	超级数字线性磁带(技术)
SDR	single data rate	单数据速率
SDRAM	synchronous DRAM	同步动态随机存储器(芯片)
SDRC	Structural Dynamics Research Corp.	结构动态研究公司(美国)
SDS	space division switching	空分交换机
SDTS	syntax-directed translation schema	语法导向翻译模式
SDU	service data unit	服务数据单元
SE	software engineering	软件工程
SE-CMM	capability maturity model for system engineering	系统工程能力成熟度模型
SEE	software engineering environment	软件工程环境
SEED	self-electrooptic effect device	自电光效应器件
SEQUEL	structured English query language	结构化的英语查询语言
SET	Standard d' Echange et de transfer	(法国和欧洲)工程数据交换标准
SETL	Set-Oriented Language	面向集合的语言, SETL 语言

缩略语	英文全名	中文译名
SGLDM	spatial grey-level dependence matrix	空间灰度关系矩阵
SGML	standard generalized markup language	标准通用置标语言
SIG	special interest group	专项组,专业组
SIGOA	Special Interest Group on Office Automation	办公自动化专业组(美国计算机协会)
SIGOIS	Special Interest Group on Office Information System	办公信息系统专业组(美国计算机协会)
SILS	standard for interoperable local area network security	互操作局域网的安全标准
SIM	subscribe identity module	用户身份组件
SIMD	single-instruction stream multiple-data stream	单指令流多数据流
SIMULA	simulation language	SIMULA[仿真]语言
SIP	SMDS interface protocol	SMDS 接口协议
SISD	single-instruction stream single-data stream	单指令流单数据流
SL	specification language	规约语言
SLAM	simulation language for analogue modeling	模拟建模的仿真语言
SLIP	serial line internet protocol	串行线路网际协议
SLM	space laser modulator	空间光调制器
SM	semiconductor memory	半导体存储器
SMAP	system management application protocol	系统管理应用协议
SMART	system for the mechanical analysis and retrieval of text	SMART 系统
SMD	storage module drive	存储模块驱动器(接口),SMD 接口
SMD	surface mounted device	表面安装器件
SMDS	switched multimegabit data service	交换式多兆位数据业务
SME	Society of Manufacturing Engineer	制造工程师协会(美国)
SMF	single mode fiber	单模光纤
SMI	structure of management information	管理信息结构
SMI	system management interrupt	系统管理中断
SMIL	Synchronized Multimedia Integration Language	同步多媒体集成语言
SML	standard metalanguage	标准元语言
SML	system message language	系统消息语言
SMM	system management mode	系统管理模式
SMPS	switching mode power supply	开关电源
SMP	symmetric multiprocessor	对称式多处理机
SMT	station management	(FDDI)站管理
SMT	surface mounting technology	表面安装技术
SMTP	Simple Mail Transfer Protocol	简单邮件传送协议

缩略语	英文全名	中文译名
SNA	system network architecture	系统网络体系结构
SNI	subscriber network interface	用户网络接口
SNIA	Storage Networking Industry Association	存储网络产业协会
SNMP	Simple Network Management Protocol	简单网络管理协议
SNOBOL	String-Oriented Symbolic Language	面向串的符号语言, SNOBOL 语言
SNRZ	synchronized non-return-to-zero	同步不归零(制)
SOC	system on a chip	片上系统
SOI	silicon-on-insulator	绝缘体上硅,单晶硅薄膜
SON	synchronous optical network	同步光纤网
SONET	synchronous optical network	同步光纤网
SOPC	system on programmable chip	可编程的片上系统
SP	stack pointer	栈指针
SP	structured programming	结构化程序设计
SPARC	scalable processor architecture	可缩放处理机体系结构
SPARC	Standards Planning and Requirements Committee (ANSI)	规划与需求标准委员会(美国国家 标准学会)
SPC	set-point control	设定值控制
SPC	stored-program control	存储程序控制
SPEC	System Performance Evaluation Cooperative	系统性能评价合作组织
SPI	serial peripheral interface	串行外围接口
SPICE	Simulation Program With Integrated Circuit Emphasis	(程序名)
SPICE	Software Process Improvement and Capacity determine	软件过程改进和能力确定(项目)
SPLICE	Simulation Program With Large Scale Integrated Circuit Emphasis	(程序名)
SPMD	single program over multiple data streams	单程序多数据流
SPOOF	structure and parity-observing output function	结构与奇偶性观察输出功能(方 法),SPOOF 法
SPOOL	simultaneous peripheral operations online	假脱机[操作]
SQL	structured query language	结构化查询语言
SRAM	static random access memory	静态随机存储器
SRB	source-routebridging	源路由选择网桥
SRI	Stanford Research Institute	斯坦福研究所(美国)
SRS	source-routeswitching	源路由选择交换
SRT	source-routetransparent	源路由透明(网桥)
SS	self-scheduling	自调度
SS	signalling system	信令系统

缩略语	英文全名	中文译名
SSA	serial storage architecture	串行存储体系结构
SSA	structured system analysis	结构化系统分析
SSAP	source services access point	源服务访问点
SSD	solid state disc	固态硬盘
SSI	small scale integration	小规模集成[电路]
SSL	secure sockets layer	安全套接字协议层
SSRAM	synchronous SRAM	同步静态随机存取存储器(芯片)
SS-TDMA	satellite switched time-division multiple access	卫星交换的时分多址
STARS	software technology for adaptable reliable system	自适应可靠系统的软件技术
STDBUS	standard bus	标准总线
STDM	statistic time division multiplexing	统计时分多路复用
STE	signalling terminal equipment	信号传输终端设备
STEP	standard for the exchange of product model data	产品模型数据交换标准
STMs	synchronous transport modules	同步传输模块
STM	synchronous transfer mode	同步传送模式
STP	shielded twisted pair	屏蔽双绞线
STRADIS	structured analysis, design and implementation of information systems	信息系统结构化分析、设计及实现
STSS	synchronous transport signals	同步传输信号
SVC	switched virtual circuit	交换[型]虚电路
SVCD	super video CD	超级 VCD 视盘
SVG	scalable vector graphic	可缩放向量图形
SVID	System V Interface Definitions	(UNIX)系统 V 界面(标准)
SVM	support vector machine	支持向量机
SW-CMM	capability maturity model for software	软件能力成熟度模型
TAB	tape automated bonding	载带自动焊
TAPI	telephone application programming interface	电话应用编程接口
TAP	TCP/IP access procedure	TCP/IP 接入规程
TBM	terabit memory	太位存储器
TC	technical committee	技术委员会
TC	terminal controller	终端控制器
TC	transmission convergence	传输会聚
TCB	trusted computing base	可信计算基
TCM	thermal conduction module	导热模块
TCM	trellis coded modulation	网格编码调制
TCP	Transmission Control Protocol	传输控制协议

缩略语	英文全名	中文译名
TCSEC	trusted computer system evaluation criteria	可信计算机系统评估准则
TCSP	theory of CSP	通信顺序进程理论
TDI	trusted database interpretation of the TCSEC	可信数据库指南
TDMA	time division multiple access	时分多址接入
TDM	time division multiplexing	时分多路复用
TDNN	time delay neural network	时间延迟神经网络
TDR	time-domain reflectometer	时域反射仪
TDS	time division switching	时分交换机
TE	terminal equipment	终端设备
TEMPEST	technique of electro-mechanical protection against emission and spurious transmission	防信息泄漏技术
TFT	thin film transistor	薄膜晶体管
TFTP	trivial file transfer protocol	普通文件传送协议
TLB	translation lookaside buffer	转换检测缓冲器
TLS	transport layer security	运输层安全(协议)
TMN	Telecommunication Management Network	电信管理网络
TMR	triple modular redundancy	三模冗余
TMS	truth maintenance system	真值维护系统
TNI	trusted network interpretation of the TCSEC	可信网络指南
TOP	technical and office protocol	技术和办公协议
TP	theoretical peak	理论峰值
TPC	Transaction Processing Council	事务处理委员会
TPDDI	twisted pair distributed data interface	双绞线分布式数据接口
TPS	transactions per second	事务处理(数)每秒
TRC	transversal redundancy check	纵横冗余检验,行列冗余检验
TRS	text retrieval system	正文检索系统
TSP	throng software process	群组软件过程
TSS	trapezoid self-scheduling	梯形[自]调度
TST	time-space-time	时间槽交换
TTCN	tree and tabular combined notation	协议测试用形式描述语言
TTL	transistor transistor logic	晶体管晶体管逻辑[电路]
TTS	text to speech	文语转换
UA	user agent	用户代理
UAL	user agent sublayer	用户代理子层
UBR	unspecified bit rate	未指定比特率(服务)
UBS	uninterruptable battery system	不间断电池系统
UCR	undercolor removal	基色去除函数

缩略语	英文全名	中文译名
UCS	Universal Multiple-Octet Coded Character Set	通用多八位编码字符集
UDM	uniform data model	统一数据模型
UDP	User Datagram Protocol	用户数据报协议
UGIS	urban geographic information system	城市地理信息系统
UI	UNIX International	UNIX 国际
UIDE	user interface development environment	用户界面开发环境
UIDT	user interface development tool	用户界面开发工具
UIL	user interface language	用户界面语言
UIMS	user interface management system	用户界面管理系统
UIMT	user interface management tool	用户界面管理工具
UIS	user interface system	用户界面系统
UITB	user interface tool box	用户界面工具箱
UITK	user interface toolkit	用户界面工具箱
ULSI	ultralarge scale integration	特大规模集成[电路]
UMA	uniform memory access	均匀存储器访问
UNC	University of North Carolina	北卡大学
UN / EDIFACT	United Nations /electronic data interchange for admin- istration,commerce and transport	联合国用于行政、商业和运输的电子数据交换(标准),联合国 EDI-FACT(标准)
UNI	ATM network node interface	ATM 网间接口
UNI	user-network interface	用户-网络接口
UPC	universal product code	通用产品代码
UPS	uninterruptable power system	不间断电源
URI	uniform resource identifier	统一资源定位标记
URL	universal resource locator	统一资源定位器
USB	universal serial bus	通用串行总线
UTF	UCS transformation format	UCS 变形显现形式
UTM	universal transverse Mercator	UTM(坐标系统)
UTP	unshielded twisted pair	无屏蔽双绞线
UVE-PROM	ultraviolet EPROM	紫外线可擦可编程只读存储器,紫外线 EPROM
VAG	visual attribute grammar	可视属性文法
VAL	a value-oriented algorithmic language	面向值的算法语言
VAL	value algorithmic language	单赋值算法语言
VC	virtual call	虚呼叫
VC	virtual channel	虚通道
VC	virtual circuit	虚电路

缩略语	英文全名	中文译名
VCD	video CD	VCD 视盘
VCI	virtual channel identifier	虚通道标识符
VCO	voltage controlled oscillator	压控振荡器
VDC	virtual device coordinate space	虚拟的设备坐标空间
VDL	Vienna defination language	维也纳定义语言
VDM	Vienna development method	维也纳开发方法
VESA	Video Electronic Standards Association	视频电子标准协会
VF	video floppy	视频软[磁]盘
VGA	video graphics array	视频图形阵列
VHDL	VHSIC hardware description language	超高速集成电路硬件描述语言
VHDSL	very high bit rate digital subscriber line	甚高速率数字用户专用线
VHLL	very high level language	甚高级语言
VHSIC	very high speed integrated circuit	超高速集成电路
VL Bus	VESA local bus	VL[局部]总线
VLIW	very long instruction word	超长指令字
VLM	virtual Lisp machine	虚拟 Lisp 机
VLR	visitor location register	访问位置寄存器
VLSI	very large scale integration	超大规模集成[电路]
VMD	virtual manufacturing device	虚拟制造设备
VME	virtual machine environment	虚拟机环境
VMS	Virtual Memory Operating System	虚拟存储器操作系统
VODER	voice demonstrator	电子言语合成器
VP	vector processor	向量处理机
VP	verb phrase	动词短语
VP	virtual path	虚通路
VPI	virtual path identifier	虚通路标识符
VPN	virtual private network	虚拟专网
VR	virtual reality	虚拟现实
VRAM	video random access memory	视频随机存取存储器
VRC	vertical redundancy check	垂直冗余检验
VRML	Virtual Reality Modeling Language	虚拟现实建模语言
VRTX	versatile real-time executive	通用实时管理程序
VSAT	very small aperture satellite terminal	甚小孔径卫星终端
VT	virtual terminal	虚拟终端
VTP	virtual terminal protocol	虚[拟]终端协议
VTR	video tape recorder	磁带录像机
VTs	virtual terminal service	虚[拟]终端服务

缩略语	英文全名	中文译名
WAE	wireless application environment	无线应用环境
WAIS	wide area information server	广域信息服务系统
WAM	Warren abstract machine	Warren 抽象机
WAN	wide area network	广域网
WAP	wireless application protocol	无线应用协议
WARC	World Administrative Radio Conference	世界无线电行政会议
WBS	work breakdown structure	工作分解结构
WC	world coordinate system	世界坐标系
W3C	World Wide Web Consortium	万维网联盟
WDM	wavelength division multiplexing	波分多路复用
WDP	Wireless Datagram Protocol	无线数据报协议
WEP	wired equivalent privacy	有线相等的秘密
WG	working group	工作组
WIMP	window-icon-menu-pointing device	WIMP 设备, WIMP(技术)
WLP	wafer level package	圆片级封装
WMF	Windows Metafile format	Windows 元文件格式
WORM	write once read many	一写多读(光盘)
WOSA	Windows open system architecture	Windows 开放系统结构
WP	weakest precondition	最弱前置条件
WPAN	wireless personal area network	无线私人网
WSI	wafer scale integration	圆片规模集成(电路)
WSP	wireless session protocol	无线会话协议
WTLS	wireless transport layer security	无线运输层安全(协议)
WTP	wireless transaction protocol	无线事项处理协议
WWW	world wide web	万维网
WYSIWIS	what you see is what I see	你所见即我所见
XDR	external data representation	外部数据表示
XIP	execute in place	就地执行
XML	extensible markup language	可扩展置标语言
XNOS	experimental network operating system	实验性网络操作系统
XPG	X-Open portability guideline	X - Open 可移植性指南
XPRS	a data extraction, processing and restructuring system	一种数据析取、处理和重构系统
XTI	X-Open transmission interface	X - Open 传输接口
YACC	yet another compiler-compiler	另一种编译程序的编译程序
ZBR	zone bit recording	区位记录
ZCAV	zone constant angular velocity	分区恒角速度
ZM	zero modulation	零调制(码)

附录 II 计算机及相关学科科技期刊

中国期刊

期刊名称	主办单位	国际刊号 ISSN
计算机研究与发展*	中国科学院计算技术研究所	1000—1239
计算机学报*	中国计算机学会, 中科院计算技术研究所	0254—4164
计算机科学与技术(英文)*	中国计算机学会, 中科院计算技术研究所	1000—9000
计算机工程与设计*	中国航天工业总公司 706 所	1000—7024
计算机工程与科学*	国防科技大学计算技术研究所	1007—130X
计算机应用*	中国科学院成都计算机应用研究所	1001—9081
计算机应用研究*	四川省计算机应用研究中心	1001—3695
计算机工程与应用*	华北计算技术研究所	1002—8331
计算机技术*	华北计算技术研究所	1002—8846
计算机应用与软件*	上海计算所, 上海软件中心	1000—386X
计算机辅助设计与图形学学报*	中国计算机学会	1003—9767
软件学报*	中国科学院软件研究所	1000—9825
工业控制计算机*	江苏省计算技术研究所	1001—182X
微电子学与计算机*	中国航天科技集团公司, 西安微电子技术研究所	1000—7180
小型微型计算机系统*	中国科学院沈阳计算技术研究所	1000—1220
微机发展*	中国计算机学会微机专委会陕西省计算机学会	1005—3751
数据通信*	信息产业部数据通信技术研究所	1002—5057
微型机与应用*	信息产业部电子第六研究所	1001—1927
CT 理论与应用研究*	中国地震局地震科学联合基金会	1004—4140
微型计算机*	科技部西南信息中心	1002—140X
软件世界	中国软件行业协会, 中国电子信息产业发展研究院	1005—2348
数值计算与计算机应用	中国科学院计算数学与科学工程计算研究所	1000—3266
微计算机应用	中国科学院声学研究所	1003—1944
计算机与信息处理标准化	全国计算机与信息处理标准化技术委员会	1002—8307
中文信息学报	中国中文信息学会, 中国科学院软件研究所	1003—0077
微电脑世界	中国计算机世界出版服务公司	1006—8708

* 为中国计算机学会会刊。

期刊名称	主办单位	国际刊号 ISSN
计算数学	中国科学院计算数学与科学工程计算研究所	0254—7791
软件	中国电子学会天津电子学会	1003—6970
微小型计算机开发与应用	天津市电子计算机研究所	1001—8786
微处理机	东北微电子研究所	1002—2279
计算机工程	华东计算技术研究所、上海计算机学会	1000—3428
电子计算机外部设备	信息产业部第 52 研究所	1004—6313
计算机科学	国家科技部西南信息中心	SN51—1239/TP (国内刊号)
计算机时代	浙江省计算技术研究所等	1006—8228
计算机应用文摘	国家科技部西南信息中心	1002—1353
中国计算机用户	《中国计算机用户》杂志社	1003—031X
电脑学习	哈尔滨工业大学等	1002—2422
福建电脑	福建省计算中心等	1002—9613
电子与电脑	电子工业出版社	1000—1077
计算机与现代化	江西省计算技术研究所等	1006—2475
模式识别与人工智能	中国自动化学会, 国家智能计算机研究开发中心	1003—6059
计算技术与自动化	中国自动化学会等	1003—6199
计算机辅助工程	上海海运学院	1006—0871
电子展望与决策	信息产业部基础产品发展研究中心	1005—846X
计算机网络世界	中国计算机用户协会网络分会	
电脑开发与应用	北方自动控制技术研究所	1003—5850
微计算机信息	中国计算机用户协会自动化分会	1000—7016
新浪潮 电脑·信息	山东省计算机用户协会等	1002—2244
中国图像图形学报	中国图像图形学会等	1006—8961
信息与电脑	北京电子信息集团	1003—9767
信息与控制	中国科学院沈阳自动化研究所	1002—0411
信息系统工程	国家信息中心, 天津市信息中心	1001—2362
计算机自动测量与控制	计算机自动测量与控制技术协会	1007—0257
计算机系统应用	中国科学院软件研究所	1003—3254
多媒体世界	国家信息中心	1005—2879
CAD/CAM 与制造业信息化	机械工业信息研究院	1005—8990
电脑爱好者	中国科学院计算技术研究所	1005—0043
大众软件	中国科学技术情报学会	1007—0060
计算机仿真	中国航天科工集团公司第 17 研究所	1006—9438
微型电脑应用	上海微型电脑应用学会	CN31—1634/TP (国内刊号)
电脑与信息技术	湖南电子研究所	1005—1228
电脑编程技巧与维护	中国信息产业商会	1006—4052
电脑	中国软件行业协会、广东省计算机用户协会	1002—9613

外国期刊

期刊名称	出版单位或主办单位	国际刊号 ISSN
<i>ACM Computing Surveys</i>	ACM Press	0360—0300
<i>ACM Letters on Programming Languages and Systems</i>	ACM Press	1057—4514
<i>ACM SIGACT News</i>	ACM Press	0163—5700
<i>ACM SIGADA Ada Letters</i>	ACM Press	0736—721X
<i>ACM SIGAPL APL Quote Quad</i>	ACM Press	0163—6006
<i>ACM SIGARCH Computer Architecture News</i>	ACM Press	0163—5964
<i>ACM SIGART Bulletin</i>	ACM Press	0163—5719
<i>ACM SIGCAS Computers and Society</i>	ACM Press	0095—2737
<i>ACM SIGCHI Bulletin</i>	ACM Press	0736—6906
<i>ACM SIGCOMM Computer Communication Review</i>	ACM Press	0146—4833
<i>ACM SIGCPR Computer Personnel</i>	ACM Press	0160—2497
<i>ACM SIGCPR Newsletter</i>	ACM Press	0160—2497
<i>ACM SIGCUE Outlook</i>	ACM Press	0163—5735
<i>ACM SIGFORTH Newsletter</i>	ACM Press	1047—4544
<i>ACM SIGGRAPH Computer Graphics</i>	ACM Press	0097—8930
<i>ACM SIGIR Forum</i>	ACM Press	0163—5840
<i>ACM SIGMALL-PC Notes</i>	ACM Press	0163—5816
<i>ACM SIGMETRICS Performance Evaluation Review</i>	ACM Press	0163—5999
<i>ACM SIGMIS Database</i>	ACM Press	0095—0033
<i>ACM SIGMOD Record</i>	ACM Press	0163—5808
<i>ACM SIGNUM Newsletter</i>	ACM Press	0163—5778
<i>ACM SIGOIS Bulletin</i>	ACM Press	
<i>ACM SIGOPS Operating Systems Review</i>	ACM Press	0163—5980
<i>ACM SIGPLAN Fortran Forum</i>	ACM Press	
<i>ACM SIGPLAN Lisp Pointers</i>	ACM Press	1045—3563
<i>ACM SIGPLAN Notices</i>	ACM Press	0362—1340
<i>ACM SIGSAC Review</i>	ACM Press	
<i>ACM SIGSAM Bulletin</i>	ACM Press	0163—5824
<i>ACM SIGSIM Simulation Digest</i>	ACM Press	0163—6103
<i>ACM SIGSOFT Software Engineering Notes</i>	ACM Press	0163—5948

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>ACM SIGUCCS Newsletter</i>	ACM Press	0736—6892
<i>ACM Transactions on Computer Systems</i>	ACM Press	0734—2071
<i>ACM Transactions on Computer-Human Interaction</i>	ACM Press	1073—0516
<i>ACM Transactions on Database Systems</i>	ACM Press	0362—5915
<i>ACM Transactions on Graphics</i>	ACM Press	0730—0301
<i>ACM Transactions on Information Systems</i>	ACM Press	1046—8188
<i>ACM Transactions on Mathematical Software</i>	ACM Press	0098—3500
<i>ACM Transactions on Modeling and Computer Simulation</i>	ACM Press	1049—3301
<i>ACM Transactions on Programming Languages and Systems</i>	ACM Press	0164—0925
<i>ACM Transactions on Software Engineering and Methodology</i>	ACM Press	1049—331X
<i>Acta Informatica</i>	Springer-Verlag New York, Inc. Journal Fulfillment Services Dept.	0001—5903
<i>Adaptive Behavior</i>	MIT Press	1059—7123
<i>Advances in Applied Mathematics</i>	Academic Press, Inc.	0196—8858
<i>Advances in Engineering Software</i>	Elsevier Science Ltd.	0965—9978
<i>AI & Society</i>	Springer-Verlag New York, Inc.	0951—5666
<i>AI Expert (Artificial Intelligence)</i>	Miller Freeman Publications Co.	0888—3785
<i>AI Magazine</i>	American Association for Artificial Intelligence	0738—4602
<i>Algorithmica; An International Journal in Computer Science</i>	Springer-Verlag New York, Inc.	0178—4617
<i>Analog Integrated Circuits and Signal Processing</i>	Kluwer Academic Publishers	0925—1030
<i>Applied Artificial Intelligence</i>	Taylor & Francis/Hemisphere	0883—9514
<i>Applied Mathematics and Computation</i>	Elsevier Science, Inc.	0096—3003
<i>Applied Networks Report</i>	International News Group	0899—5680
<i>Applied Numerical Mathematics</i>	Elsevier Science Publishers B. V. The Netherlands	0168—9274
<i>Artificial Intelligence</i>	Elsevier Science Publishers B. V. The Netherlands	0004—3702
<i>Artificial Intelligence and Law</i>	Kluwer Academic Publishers	0924—8463
<i>Artificial Intelligence Review</i>	Kluwer Academic Publishers	0269—2821
<i>Artificial Life</i>	MIT Press	1064—5462
<i>ASLIB Proceedings</i>	The Association for Information Management (UK)	0001—253X

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Australian Computer Journal</i>	Australian Computer Society, Inc.	0004—8917
<i>Automated Software Engineering</i>	Kluwer Academic Publishers	
<i>The Bulletin of Symbolic Logic</i>	Association for Symbolic Logic, Inc.	1079—8986
<i>Byte</i>	McGraw-Hill Publication, Inc.	0360—5280
<i>C++ Report</i>	SIGS Publications, Inc.	1040—6042
<i>C/C++ Users Journal</i>	R & D Publications, Inc.	1075—2838
<i>Canadian Information Processing/ Informatique Canadienne</i>	Canadian Information Processing Society	1182—3097
<i>CASE Trends(Computer-Aided Software Engineering)</i>	Software Productivity Group, Inc.	1046—5944
<i>CD-ROM World</i>	Mecklermedia Corp.	1066—274X
<i>Cipher</i>	IEEE Press	
<i>Circuits, Systems, and Signal Processing</i>	Birkhäuser Boston, Inc.	0278—081X
<i>Client/Server Computing</i>	Sentry Publishing Company, Inc.	1059—3470
<i>Client/Server Today</i>	Cahners Publishing Company	1078—8565
<i>Collegiate Microcomputer</i>	Rose-Hulman Institute of Technology	0731—4213
<i>Communications of the ACM</i>	ACM Press	0001—0782
<i>Computational Complexity</i>	Birkhäuser Boston, Inc.	1016—3328
<i>Computational Economics</i>	Kluwer Academic Publishers	0927—7099
<i>Computational Geometry: Theory and Applications</i>	Elsevier Science Publishers B. V. The Netherlands	0925—7721
<i>Computational Linguistics</i>	MIT Press	0891—2017
<i>Computational Mathematics and Mathematical Physics</i>	Elsevier Science, Inc.	0965—5425
<i>Computational Optimization and Applications</i>	Kluwer Academic Publishers	0926—6003
<i>Computational Statistics & Data Analysis</i>	Elsevier Science Publishers B. V. The Netherlands	0167—9473
<i>Computer</i>	IEEE Computer Society Press	0018—9162
<i>Computer Aided Design</i>	Butterworth-Heinemann Turpin Transactions Ltd. (UK)	0031—3202
<i>Computer-Aided Engineering</i>	Penton Business Publishing	0733—3536
<i>Computer Aided Geometric Design</i>	Elsevier Science Publishers B. V. The Netherlands	0167—8396
<i>Computer Design</i>	Penn Weli Publishing Co.	0010—4566
<i>Computer Fraud and Security</i>	Elsevier Advanced Technology(UK)	1361—3723
<i>Computer Graphics Forum</i>	Blackwell Publishers	0167—7055
<i>The Computer Journal</i>	Oxford University Press(UK)	0010—4620
<i>Computer Languages</i>	Pergamon Press, Inc. Imprint of Elsevier Science Inc.	0096—0551
<i>Computer Language</i>	Computer Language	0747—2839

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Computer literature Index</i>	Applied Computer Research, Inc.	0270—4846
<i>Computer Networks and ISDN Systems</i>	Elsevier Science Publishers B. V. The Netherlands	0169—7552
<i>Computer Processing of Chinese and Oriental Languages</i>	USA	0715—9048
<i>Computer Optimization and Applications</i>	Kluwer Academic Publishers	0926—6003
<i>Computer Pictures</i>	Montage Publishing, Inc.	0883—5683
<i>Computer Report</i>	日本经营科学研究所	0385—6658
<i>Computer Science and Informatics</i>	Computer Society of India	0254—7813
<i>Computer Security Journal</i>	Computer Security Institute	0277—0865
<i>Computer Speech and Language</i>	Harcourt Brace Jovanovich Ltd. (UK)	0885—2308
<i>Computer Standards and Interfaces</i>	Elsevier Science Publishers B. V. The Netherlands	0920—5489
<i>Computer Supported Cooperative Work</i>	Kluwer Academic Publishers	0925—9724
<i>Computer Systems Science and Engineering</i>	CRL Publishing Ltd. (UK)	0267—6192
<i>Computer Vision and Image Understanding</i>	Academic Press, Inc.	1077—3142
<i>Computers and Artificial Intelligence</i>	Inst. of Computer Systems	0232—0274
<i>Computers and Biomedical Research</i>	Academic Press, Inc.	0010—4809
<i>Computers and Education</i>	Pergamon Press, Inc. Imprint of Elsevier Science Inc.	0360—1315
<i>Computers and Electrical Engineering</i>	Pergamon Press, Inc. Imprint of Elsevier Science Inc.	0045—7906
<i>Computers and Fluids</i>	Pergamon Press, Inc. Imprint of Elsevier Science Inc.	0045—7930
<i>Computers and Geosciences</i>	Pergamon Press, Inc. Imprint of Elsevier Science Inc.	0098—3004
<i>Computers and Graphics</i>	Elsevier Science Publishers	0097—8493
<i>Computers and Industrial Engineering</i>	Pergamon Press, Inc. Imprint of Elsevier Science Inc.	0360—8352
<i>Computers and Operations Research</i>	Pergamon Press, Inc. Imprint of Elsevier Science Inc.	0305—0548
<i>Computers and Security</i>	Elsevier Advanced Technology Publications(UK)	0167—4048
<i>Computers in Human Services</i>	Haworth Press, Inc.	0740—445X
<i>Computers in Industry</i>	Elsevier Science Publishers B. V. The Netherlands	0166—3615
<i>Computers in Physics</i>	American Institute of Physics, Inc.	0894—1866

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Computing</i>	Springer-Verlag New York, Inc.	0010—485X
<i>Computing Reviews</i>	ACM Press	0010—4884
<i>Computing Systems</i>	MIT Press	0895—6340
<i>Data & Knowledge Engineering</i>	Elsevier Science Publishers B. V. The Netherlands	0169—023X
<i>Database</i>	Online, Inc.	0162—4105
<i>Database and Network Journal</i>	A. P. Publications Ltd.	0308—3314
<i>Data Base Product Report</i>	Management Information Corp.	0740—6800
<i>Database Programming & Design</i>	Miller Freeman Publications	0895—4518
<i>Data Communications International</i>	McGraw Hill Publication, Inc.	
<i>Datamation</i>	Cahners Publishing Company	0011—6963
<i>DBMS</i>	Miller Freeman Publications	1041—5173
<i>Decision Support Systems</i>	Elsevier Science, Inc.	0167—9236
<i>Discrete Applied Mathematics</i>	Elsevier Science Publishers B. V. The Netherlands	0166—218X
<i>Discrete Mathematics</i>	Elsevier Science Publishers B. V. The Netherlands	0012—365X
<i>Distributed and Parallel Databases</i>	Kluwer Academic Publishers	0926—8782
<i>Distributed Computing</i>	Springer-Verlag(Germany)	0178—2770
<i>Distributed Systems Engineering Journal</i>	IOP Publishing Ltd. , (UK)	0967—1846
<i>Electronic Publishing—Origina- tion, Dissemination, and Design</i>	John Wiley and Sons Ltd.	0894—3982
<i>Expert Systems</i>	Learned Information Ltd.	
<i>Finite Elements in Analysis and Design</i>	Elsevier Science Publishers B. V. The Netherlands	0168—874X
<i>Formal Methods in System Design</i>	Kluwer Academic Publishers	0925—9856
<i>Future Generation Computer Systems</i>	Elsevier Science Publishers B. V. The Netherlands	0167—739X
<i>Fuzzy Sets and Systems</i>	Elsevier Science Publishers B. V. The Netherlands	0165—0114
<i>Graphical Models and Image Processing</i>	Academic Press, Inc.	1077—3169
<i>Home Computer Magazine</i>	Emerald Valley Publishing Co.	0747—055X
<i>IBM Journal of Research and Development</i>	IBM Corp.	0018—8646
<i>IBM Systems Journal</i>	IBM Corp.	0018—8670
<i>IEE Proceedings—Computers and Digital Techniques</i>	IEE	1350—2387
<i>IEE Proceedings—Vision</i>	Image and Signal Processing IEE	

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>IEEE/ACM Transactions on Networking</i>	ACM Press	1063—6692
<i>IEEE Annals of the History of Computing</i>	IEEE Computer Society Press	1058—6180
<i>IEEE Computational Science & Engineering</i>	IEEE Computer Society Press	1070—9924
<i>IEEE Computer Graphics and Applications</i>	IEEE Computer Society Press	0272—1716
<i>IEEE Design & Test</i>	IEEE Computer Society Press	0740—7475
<i>IEEE Micro</i>	IEEE Computer Society Press	0272—1732
<i>IEEE MultiMedia</i>	IEEE Computer Society Press	1070—986X
<i>IEEE Network</i>	IEEE Press	0890—8044
<i>IEEE Parallel & Distributed Technology: Systems & Technology</i>	IEEE Computer Society Press	1063—6552
<i>IEEE Software</i>	IEEE Computer Society Press	0740—7459
<i>IEEE Spectrum</i>	IEEE Press	0018—9235
<i>IEEE Transactions on Computers</i>	IEEE Press	0018—9340
<i>IEEE Transactions on Parallel & Distributed Systems</i>	IEEE Press	1045—9219
<i>IEEE Transactions on Pattern Analysis and Machine Intelligence</i>	IEEE Press	0162—8828
<i>IEEE Transactions on Software Engineering</i>	IEEE Press	0098—5589
<i>IEEE Transactions on Very Large Scale Integration (VLSI) Systems</i>	IEEE Press	1063—8210
<i>IFIP Transaction, A: Computer Science and Technology</i>	Elsevier Science Publishers	0926—5473
<i>Image Understanding</i>	Academic Press	1049—9660
<i>IMPACT of Computing in Science and Engineering</i>	Academic Press, Inc.	0899—8248
<i>Information and Computation</i>	Academic Press, Inc.	0890—5401
<i>Information and Management</i>	Elsevier Science, Inc.	0378—7206
<i>Information Processing and Management: An International Journal</i>	Pergamon Press, Inc. Imprint of Elsevier Science Inc.	0306—4573
<i>Information Processing Letters</i>	Elsevier Science Publishers B. V. The Netherlands	0020—0190
<i>Information Resources Management Journal</i>	Idea Group Publishing	1040—1628

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Information Sciences—Applications; An International Journal</i>	Elsevier Science, Inc.	1069—0115
<i>Information Sciences—Informatics and Computer Science; An International Journal</i>	Elsevier Science, Inc.	0020—0255
<i>Information Sciences—Intelligent Systems; An International Journal</i>	Elsevier Science, Inc.	0020—0255
<i>Information Systems</i>	Pergamon Press, Inc.	0306—4379
<i>Integration, the VLSI Journal</i>	Elsevier Science Publishers B. V. The Netherlands	0167—9260
<i>Intelligent Software Strategies</i>	Cutter Information Corp.	0887—221X
<i>Interacting with Computers</i>	Butterworth-Heinemann	0953—5438
<i>International Journal in Computer Simulation</i>	Ablex Publishing Corp.	1055—8470
<i>International Journal of Applied Software Technology</i>	International Academic Publishing Co. (Canada)	1198—5577
<i>International Journal of Computer Mathematics</i>	Gordon and Breach Science Publishers	0027—7160
<i>International Journal of Computer Systems</i>	CRL	
<i>International Journal of Computer Systems Science and Engineering</i>	CRL	
<i>International Journal of Computer Vision</i>	Kluwer Academic Publishers	0920—5691
<i>International Journal of Expert Systems</i>	JAI Press, Inc.	0894—9077
<i>International Journal of Foundations of Computer Science</i>	World Scientific Publishing Co. (Singapore)	0129—0541
<i>International Journal of Game Theory</i>	Springer-Verlag New York, Inc. Journal Fulfillment Services Dept.	0020—7276
<i>International Journal of High-Speed Computing</i>	World Scientific Publishing Co. (Singapore)	0129—0533
<i>International Journal of Human—Computer Interaction</i>	Ablex Publishing Corp.	1044—7318
<i>International Journal of Human—Computer Studies</i>	Academic Press, Inc. (UK)	1071—5819
<i>International Journal of Intelligent Systems</i>	John Wiley and Sons, Inc.	0884—8173
<i>International Journal of Man—Machine Studies</i>	Academic Press, Inc. (UK)	0020—7373

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>International Journal of Mini and Microcomputers</i>	ISMM	0702—0481
<i>International Journal of Parallel Programming</i>	Plenum Press Imprint of Plenum Publishing Corp.	0885—7458
<i>International Journal of Pattern Recognition and Artificial Intelligence</i>	World Scientific Publishing Co. (Singapore)	0218—0014
<i>International Journal of Robotics Research</i>	MIT Press	0278—3649
<i>International Journal of Software Engineering and Knowledge Engineering</i>	World Scientific Publishing Co. (Singapore)	0218—1940
<i>International Journal of Super-computer Applications and High Performance Engineering</i>	MIT Press	0890—2720
<i>International Journal of Virtual Reality</i>	N. Cascade Avenue, Colorado Springs	1081—1451
<i>Internet Research</i>	MCB University Press	1066—2243
<i>Journal of Algorithms</i>	Academic Press, Inc.	0196—6774
<i>Journal of Artificial Intelligence in Education</i>	Association for the Advancement of Computing in Education	1043—1020
<i>Journal of Artificial Neural Networks</i>	Ablex Publishing Corp.	1073—5828
<i>Journal of Automated Reasoning</i>	Kluwer Academic Publishers	0168—7433
<i>Journal of Circuits, Systems and Computers</i>	World Scientific Publishing	
<i>Journal of Computational & Applied Mathematics</i>	Elsevier Science Publishers B. V. The Netherlands	0377—0427
<i>Journal of Computer and Software Engineering</i>	Ablex Publishing Corp.	1069—5451
<i>Journal of Computer and System Sciences</i>	Academic Press, Inc.	0022—0000
<i>Journal of Database Management</i>	Idea Group Publishing	1063—8016
<i>Journal of Educational Multimedia and Hypermedia</i>	Association for the Advancement of Computing in Education	1055—8896
<i>Journal of Electronic Testing Theory and Applications</i>	Kluwer Academic Publishers	0923—8174
<i>Journal of End User Computing</i>	Idea Group Publishing	1063—2239
<i>Journal of Experimental & Theoretical Artificial Intelligence</i>	Taylor & Francis, Inc.	0952—813X
<i>Journal of High Speed Networks</i>	IOS Press	0926—6801
<i>Journal of Graph Theory</i>	John Wiley & Sons, Inc.	0364—9024

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Journal of Information Processing</i>	日本情报处理学会	0387—6101
<i>Journal of Information Science</i>	Bowker-Saur Limited(UK)	0165—5515
<i>Journal of Intelligent Information Systems</i>	Kluwer Academic Publishers	0925—9902
<i>Journal of KISS(A) (Computer Systems and Theory)</i>	Korea Information Science Society	
<i>Journal of KISS(B) (Software and Applications)</i>	Korea Information Science Society	
<i>Journal of KISS(C) (Computing Practices)</i>	Korea Information Science Society	
<i>Journal of Logic Programming</i>	Elsevier Science, Inc.	0743—1066
<i>Journal of Management Information Systems</i>	M. E. Sharpe, Inc.	0742—1222
<i>Journal of Mathematical Imaging and Vision</i>	Kluwer Academic Publishers	0924—9907
<i>Journal of Microcomputer Applications</i>	Academic Press, Inc. (UK)	0745—7138
<i>Journal of Network and Systems Management</i>	Plenum Press Imprint of Plenum Publishing Corp.	1064—7570
<i>Journal of New Generation Computer Systems</i>	STBS Ltd.	0863—0445
<i>Journal of Parallel and Distributed Computing</i>	Academic Press, Inc.	0743—7315
<i>Journal of Scientific Computing</i>	Plenum Press Imprint of Plenum Publishing Corp.	0885—7474
<i>The Journal of Supercomputing</i>	Kluwer Academic Publishers	0920—8542
<i>Journal of Symbolic Computation</i>	Academic Press, Inc. (UK)	0747—7171
<i>Journal of Symbolic Logic</i>	Association for Symbolic Logic, Inc.	0022—4812
<i>Journal of Systems and Software</i>	Elsevier Science, Inc.	0164—1212
<i>Journal of the ACM</i>	ACM Press	0004—5411
<i>Journal of the American Society for Information Science(JASIS)</i>	John Wiley & Sons, Inc.	0002—8231
<i>Journal of the Computer Society of India</i>	Computer Society of India	0045—7892
<i>Journal of VLSI Signal Processing Knowledge Acquisition</i>	Kluwer Academic Publishers	0922—5773
<i>Knowledge-based Systems</i>	Academic Press, Inc. (UK)	1042—8143
<i>Languages of Design</i>	Butter Worth-Heinemann Turpin Transactions Ltd.	0950—7051
<i>Lisp and Symbolic Computation</i>	Elsevier Science Publishers B. V.	0927—3034
<i>Local Area Network(LAN)</i>	Kluwer Academic Publishers	0892—4635
<i>Machine Learning</i>	Information Gatekeepers, Inc.	1051—1962
<i>Machine Vision and Applications</i>	Kluwer Academic Publishers	0885—6125
	Springer-Verlag New York, Inc.	0932—8092

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Management Science</i>	The Institute for Operations Research and Management Sciences	0025—1909
<i>Mathematical Programming</i>	Elsevier Science Publishers B. V. The Netherlands	0025—5610
<i>Mathematics of Computation</i>	American Mathematical Society	0025—5718
<i>Methods of Logic in Computer Science</i>	Ablex Publishing Corp.	1075—0924
<i>Microelectronic Engineering</i>	Elsevier Science Publishers B. V. The Netherlands	0167—9317
<i>Microprocessing and Microprogramming</i>	Elsevier Science Publishers B. V. The Netherlands	0165—6074
<i>Microprocessors & Microsystems</i>	Butterworth-Heinemann	0141—9331
<i>Micro-Systems</i>	Society Parisienne d' Edition(France)	
<i>MIS Quarterly</i>	Society for Information Management and The Management Information Systems Research Center	0276—7783
<i>Multidimensional Systems and Signal Processing</i>	Kluwer Academic Publishers	0923—6082
<i>Multimedia Systems</i>	Springer-Verlag New York, Inc.	0942—4962
<i>Multimedia Systems</i>	Information Europe Ltd.	0960—7749
<i>Multimedia Tools and Applications</i>	Kluwer Academic Publishers	1380—7501
<i>Network Computing</i>	CMP Publications, Inc.	1046—4468
<i>Networks</i>	Wiley-Interscience	0028—3045
<i>Neural Computation</i>	MIT Press	0899—7667
<i>Neural Networks</i>	Pergamon Press, Inc.	0893—6080
<i>Neural, Parallel & Scientific Computations</i>	Dynamic Publisher, Inc.	1061—5369
<i>New Generation Computing</i>	Springer-Verlag New York, Inc.	0288—3635
<i>New Review of Applied Expert Systems</i>	Taylor Graham Publishing(UK)	1361—0244
<i>New Review of Hypermedia and Multimedia</i>	Applications and Research(UK)	1361—4568
<i>On The Internet</i>	Rickard Associates	1081—3969
<i>OS 2 Professional</i>	I. F. Computer Media, Inc.	1069—6814
<i>Packaged Software Report; CASE Product Report</i>	Management Information Corp.	0747—9573
<i>Parallel Algorithms and Applications</i>	STBS Ltd. Order Department	1063—7192
<i>Parallel Computing</i>	Elsevier Science Publishers B. V. The Netherlands	0167—8191
<i>Pattern Recognition</i>	Elsevier Science Ltd.	0031—3203

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Pattern Recognition Letters</i>	Elsevier Science Publishers B. V. The Netherlands	0167—8655
<i>PC Magazine</i>	Ziff. Davis Publishing Co.	0888—8507
<i>Performance Evaluation</i>	Elsevier Science Publishers B. V. The Netherlands	0166—5316
<i>Presence: Teleoperators and Virtual Environments</i>	MIT Press	1054—7460
<i>Real-Time Systems</i>	Kluwer Academic Publishers	0922—6443
<i>Resource Sharing and Information Networks</i>	Haworth Press, Inc.	0737—7797
<i>Science of Computer Programming</i>	Elsevier Science Publishers	0167—6423
<i>Scientific Programming</i>	John Wiley & Sons, Inc.	1058—9244
<i>SIAM Journal on Applied Mathe- matics</i>	Society for Industrial and Applied Mathematics	0036—1399
<i>SIAM Journal on Computing</i>	Society for Industrial and Applied Mathematics	0097—5397
<i>SIAM Journal on Control and Optimization</i>	Society for Industrial and Applied Mathematics	0353—0129
<i>SIAM Journal on Discrete Mathe- matics</i>	Society for Industrial and Applied Mathematics	0895—4801
<i>SIAM Journal on Mathematical Analysis</i>	Society for Industrial and Applied Mathematics	0036—1410
<i>SIAM Journal on Matrix Analysis and Applications</i>	Society for Industrial and Applied Mathematics	0895—4798
<i>SIAM Journal on Numerical Analysis</i>	Society for Industrial and Applied Mathematics	0036—1429
<i>SIAM Journal on Scientific Computing</i>	Society for Industrial and Applied Mathematics	1064—8275
<i>Signal Processing</i>	Elsevier Science Publishers B. V. The Netherlands	0165—1684
<i>Simulation</i>	Society for Computer Simulation	0037—5497
<i>Simulation and Gaming</i>	Sage Publications, Inc.	1046—8781
<i>Simulation Practice and Theory</i>	Elsevier Science Publishers B. V. The Netherlands	0928—4869
<i>Sixth Generation Systems</i>	Gallifrey Publishers	
<i>Social Science Computer Review</i>	Duke University Press	0894—4393
<i>Software Concepts and Tools</i>	Springer-Verlag	
<i>Software Development</i>	Miller Freeman Publications	1070—8588
<i>Software Engineering Journal</i>	Institution of Electrical Engineers(UK)	0268—6961
<i>Software—Industry Report</i>	Computer Age and EDP News Services	
<i>Software Law Journal</i>	Center for Computer/Law	
<i>Software Magazine</i>	Sentry Publishing Company, Inc.	0897—8085
<i>Software Management News</i>	Software Maintenance News, Inc.	0741—4501

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Software—Practice & Experience</i>	John Wiley & Sons, Inc.	0038—0644
<i>Software Quality Journal</i>	Chapman and Hall (UK)	
<i>Software Testing, Verification and Reliability</i>	John Wiley and Sons Ltd. (UK)	
<i>Speech Communication</i>	Elsevier Science Publishers B. V. The Netherlands	0167—6393
<i>Standard View</i>	ACM Press	1067—9936
<i>System and Computers in Japan</i>	John Wiley and Sons, Inc.	0882—1666
<i>Systems Analysis Modelling Simulation</i>	Gordon and Breach Science Publishers, Inc.	0232—9298
<i>Systems Integration Business</i>	Cahners Publishing Co.	0364—9342
<i>Theoretical Computer Science</i>	Elsevier Science Publishers B. V. The Netherlands	0304—3975
<i>Theory and Practice of Object Systems</i>	John Wiley & Sons, Inc.	1074—3227
<i>Topology and Its Applications</i>	Elsevier Science Publishers B. V. The Netherlands	0166—8641
<i>Transaction of Information Processing Society of Japan</i>	Information Processing Society of Japan	0387—5806
<i>Transactions of the Society for Computer Simulation</i>	Society for Computer Simulation	0740—6797
<i>The Visual Computer</i>	Springer-Verlag New York, Inc. Journal Fulfillment Services Dept.	0178—2789
<i>Unix Review</i>	Miller Freeman Publications Co.	0742—3136
<i>Visual Computer</i>	Springer-Verlag (Germany)	0178—2789
<i>Visualization and Computer Animation</i>	Wiley Publishers	1049—8907
<i>VLSI Design</i>	Gordon and Breach	0279—2837
<i>Windows User</i>	Wandsworth Publishing, Inc.	1065—3481
<i>Windows/DOS Developer's Journal</i>	R & D Publications, Inc.	1059—2407
<i>Wireless Networks</i>	ACM Press	1022—0038
<i>Wireless Personal Communications; An International Journal</i>	Kluwer Academic Publishers	0929—6212
<i>Wirtschafts Informatik</i>	Friedr. Vieweg & Sohn Verlags-gesellschaft mbH (Germany)	0937—6429
コンピューター ソフトウェア (Computer Software)	岩波书店情报处理, 日本情报处理学会	0447—8053
Автоматика и вычислительная техника	Международная Книга	0132—4160
Вычислительные машины и системы, выпуск "РЖ"	Международная Книга, Москва	0234—9663

期 刊 名 称	出版单位或主办单位	国际刊号 ISSN
<i>Вычислительные науки, отдельный выпуск "РЖ"</i>	Международная Книга, Москва	0235 --- 1501
<i>Вычислительная техника (ЭИ)</i>	Международная Книга, Москва	0132 — 1730
<i>Программирование</i>	Международная Книга, Москва	0132 — 3474
<i>Микроэлектроника</i>	Москва, Российская Академика Наук	0544 — 1269

主要参考资料

1. Source Publications Lists. Science Citation Index (SCI). 3D. Institute of Science Information, Inc., 1996
2. INSPEC List of Journals. Computer and Control Abstracts. Science Abstracts Series C, December, 1995, and № 1 ~ 8, 1996, INSPEC
3. Computing Reviews. November, 1995, 599 ~ 608
4. Computer Abstracts. 1995, 39: 930 ~ 931
5. 外国报刊目录. 万国学术出版社, 1993, 1995
6. 报刊简明目录. 北京报刊发行局, 1997
7. 北京图书馆期刊目录.
8. 中国科学院文献情报中心期刊目录.

(张 伟)

附录Ⅲ 计算机及相关学科学术团体

1. 中国计算机学会, China Computer Federation (CCF)

成立于1962年.

地址: 中国, 北京市海淀区中关村南4街, 100080, 北京 2704 信箱.

China, Beijing 100080, P. O. Box 2704.

电话: 86-010-62562503

传真: 86-010-62567724

E-mail: ccf @ ict. ac. cn

2. 中国自动化学会, Chinese Association of Automation (CAA)

成立于1961年11月.

地址: 中国, 北京市海淀区中关村东路95号, 100080, 北京 2728 信箱.

China, Beijing 100080, P. O. Box 2728.

电话: 86-010-62544415

传真: 86-010-62545229

E-mail: zdhxx@ hotmail. com

3. 中国通信学会, China Institute of Communication (CIC)

成立于1980年12月.

地址: 中国, 北京西长安街13号, 100804.

China, Beijing 100804.

电话: 86-010-66051385

传真: 86-010-66069587

E-mail: register@ China-cic. org

4. 中国中文信息学会, Chinese Information Processing Society of China (CIPSC)

成立于1981年6月.

地址: 中国, 北京市海淀区中关村南4街4号, 100080, 北京 8718 信箱.

China, Beijing 100080, P. O. Box 8718.

电话: 86-010-62562916

传真: 86-010-62562533

5. 中国电子学会, Chinese Institute of Electronics (CIE)

成立于1962年.

地址: 中国, 北京市海淀区玉渊潭南路普惠南里综合楼, 100036, 北京 165 信箱.

China, Beijing 100036, P. O. Box 165.

电话: 86-010-68283461

传真: 86-010-68283458

6. 中国图像图形学会, China Society of Image and Graphics

成立于1990年.

地址: 中国, 北京市海淀区花园路6号, 100088.

China, Beijing 100088.

电话: 86-010-62014411-2503

7. 香港电脑学会, Hong Kong Computer Society (HKCS)

成立于1970年.

地址: 中国, 香港湾仔石水渠街1号, 其发大厦一字楼D室.

Hong Kong, Unit D, 1/F., Luckifast Building, No. 1 Stone Nuliah Lane, Wan Chai. China.

电话: 852-2834 2228 传真: 852-2834 3004

E-mail: hkes @ hkes. org. hk

http: //www. nkes. org. hk

8. Taipei Computer Association

成立于1974年.

Add: Taipei, 3rd floor, 2 PA teh Road, section 3. China.

Tel: 2-7764249 Fax: 2-7764410

http: //www. tca. org. tw

9. American Association for Artificial Intelligence (AAAI)

Add: 445 Burgess Drive, Menlo Park, California 94025. U. S. A.

Tel: 1-(415)328-3123 Fax: 1-(415)321-4457

http: //www. aaai. org

10. American Federation of Information Processing Societies (AFIPS)

成立于1961年.

Add: 1899 Preston White Drive, Reston. VA 22091. U. S. A.

Tel: 1-(703) 620-8900

11. American Society for Information Science (ASIS)

成立于1937年. 1968年改为现名.

Add: N. W. Suite 404, Washington DC 20036. U. S. A.

Tel: 1-(202)462-1000

http: //www. asis. org

12. Associacao Portuguesa de Informatica (AIP)

Add: Portugal, Av. Almirante Reis 127, 1°. Esq. P-1100 LISBON.

Tel: 351-(1) 535587 Fax: 351-(1) 570410

Telex: 64653apint p.

http: //www. api. pt

13. Association for Computational Linguistics (ACL)

Add: PO Box 6090 Somerset, NJ 08875, U. S. A.

http: //www. cs. columbia. edu / ~acl /home. html

14. Association for Computing Machinery (ACM)

成立于1947年9月15日.

Add: 1515 Broadway, New York, NY10036. U. S. A.

Tel: 1-(212) 6260500 Fax: 1-(212) 9441318

E-mail: acmhelp @ acm. org

http: //www. acm. org

15. Association for Systems Manegement (ASM)

成立于1947年.

Add: 24587 Begley Road ,Cleveland, ON 44138. U. S. A.

Tel: 1-(216) 243-6900

http: //aww. asm. org

16. Association Francaise pour la Cybernetique Economique et Technique(AFCET)

Add: 156 boulevard Pereire, F-75017 Paris. France.

Tel: 33-(1) 47662419 Fax: 33-(1) 42679312

Telex: 283155 F Code 235 E.

17. Association of Data Processing Service Organizations (ADAPSO)

Add: 1300 North 17th Street, Arlington, VA 22209. U. S. A.

Tel: 1-(703) 522-5055

http: //www. cni. org /docs /infopols /adapso. html *

18. Associazione Italiana per L' Informatica ed il Calcolo Automatico(A. I. C. A.)

Add: Piazza R. morandi, 2, I-20121 MILAN. Italy.

Tel: 39-(2) 784970 Fax: 39-(2) 76014082

19. Australian Computer Society (ACS)

Add: P. O. Box 319, DARLINGHURST, N. S. W. 200. Australian.

Tel: 61-(2) 2115855 Fax: 61-(2) 2811208

http: //www. acs. org. au

20. Austrian Computer Society

Add: Wollzeile 1-3, A-1010 VIENNA. Austria.

Tel: 43-(1) 5120235 Fax: 43-(1) 5137735

21. Bangladesh Computer Society

成立于1979年.

Add: c /o Computer Centre, Bangladesh University of Engineering and Technology, Dhaka1000,
Bangladesh.

Tel: 880-(2)503744 Fax: 880-(2)863026

http: //www. tuns. ca / ~ abidmr /buet. html

22. British Computer Society (BCS)

成立于1957年.

- Add: P. O. Bpx 1454, Station Road, Swindon, SNI 1TG, United Kingdom.
Tel: 44-(793) 480269 Fax: 44-(793) 480270
http: //www. bcs. org. uk
23. Canadian Information Processing Society (CIPS)
Add: 430 King St. West, Suite 205, TORONTO, Ontario, Canada M5V 1L5. Canada.
Tel: 1-(416) 5934040 Fax: 1-(416) 5935184
http: //www. cips. ca
24. Centre Nacional d' Informatica d' Andorra (CNIA)
Add: Avgda. Santa Coloma, 91 ANDORRA LA VELLA. Andorra.
Tel: 33-(628) 22400 Fax: 33-(628) 28218
http: //www. andorra. ad
25. Centre National de L' Informatique
Add: 17 rue Belhassan Ben Chaabane, EL Omrane, 1005 TUNIS. Tunisia.
Tel: 216-(1) 782996 Fax: 216-(1) 781862
Telex: 13904 ceninf tn.
E-mail: kamoun @ tuniscni. uucp
26. Chinese & Oriental language Information Processing Society (COLIPS)
Add: Department of Information System and Computer Science, National University of Singapore. Kent Ridge, Singapore.
Fax: 65-(779) 4580
E-mail: luakt @ iscs. nus. sg
http: //www. iscs. nus. sg / ~ colips / commcolips
27. Computer Association of Nigeria (CAN)
Add: P. O. Box 4800, 5 Akinhanmi Street, SURULERE, LAGOS. Nigeria.
28. Computer Association of Thailand
成立于1972年。
Add: 2nd Floor, Chulalongkorn University Alumni Association Building, Phayathai Road, Bangkok 10330, Thailand.
Tel: 66-(2) 2153546 Fax: 66-(2) 2153962
http: //www. nectec. or. th
29. Computer Society of India (CSI)
成立于1964年。
Add: . c / o Institution of Engineers Building, K. Khadye Marg, 15 haji Ali Park, BOMBAY 4000 034. India.
Tel: 91-(22) 4943422 or 4949433
30. Computer Society of Pakistan

成立于1973年.

Add: Office 5, 3rd Floor, Sasi Arcade, Main Clifton Road, Karachi, Pakistan.

Tel: 92-(21) 5701030

31. Computer Society of South Africa

Add: P. O. Box 1714, HALFWAY HOUSE 1685. Rep. of South Africa. South Africa.

Tel: 27-(11) 3151319 Fax: 27-(11) 3152276

32. Computer Society of Zimbabwe

Add: P. O. Box 8385, Causeway, HARARE. Zimbabwe.

Tel: 263-(4) 706725 Fax: 263-(4) 725657

Telex: 26433 cmihre zw.

33. Czechoslovak Committee for IFIP

Add: Czechoslovakia. c/o Dr. J. Dolezal (Nat. Corres.), Czech. Academy of Sciences, Pod vodarenskou vezi 4, CS -18208 PRAGUE.

Tel: 42-(2) 8152062 Fax: 42-(2) 847452

Telex: 122018 atom c.

E-mail: dolezal @ cspgasll

34. Danish Federation for Information Processing (DANFIP)

Add: Kronprinsensgade 14, DK-1114 COPENHAGEN K. Denmark.

Tel: 45-33-111560 Fax: 45-33-931580

35. European Association for Computer Graphics

Add: EUROGRAPHICS Association, P. O. Box 16, CH-1288 Aire-la-Ville, Switzerland.

Fax: 41-(22) 7570318

E-mail: secretart @ eg.org

http: //www. et.org

36. Egyptian Computer Society

Add: P. O. Box 9009, NasrCity, CAIRO, Tth A. R. E. Egypt.

Tel: 20-(2) 601484

37. European Association for Microprocess. and Microprogramming (EUROMICRO)

Add: P. O. Box 2346, NL-7301 EA APELDOORN, Netherlands.

Tel: 31-(55) 557372 Fax: 31-(55) 557393

E-mail: eruomicro @ standby.nl

http: //europub. ict. tuwien. ac.at

38. FAIB -FBVI

Add: Square de Biarritz 3, Bte 5, B -1050 BRUSSELS. Belgium.

Tel: 32-(2) 2371045 Fax: 32-(2) 2306785

Telex: 22075.

http: //www. bfia. be

39. Federation Espanola de Sociedades de Informatica (FESI)

Add: Hortaleza 104 -2º. izqda. E-28004 MADRID. Spain.

Tel: 34-(1) 5192565 Fax: 34-(1) 5440763

E-mail: fesi@ dip. upm. es

http: //www. upm. es *

40. Federation on Computing in the United States (FOCUS)

1991 年由 ACM 和 IEEE CS 联合组成, 作为 IFIP 的成员.

Add: 1730 Massachusetts Aenue, N. W. WASHINGTON, D. C. 20036-1903. U. S. A.

Tel: 1-(202) 3710101 Fax: 1-(202) 7289614

41. Finnish Information Processing Association

Add: P. O. Box 68, SF-02601 ESPOO. Finland.

Tel: 358-(90) 5121255 Fax: 358-(90) 5121276

http: //www. ttlry. fi /2english. htm

42. Gesellschaft fur Informatik (GI)

成立于 1962 年.

Add: Godesberger Allee 99, D -5300 BONN 2. Germany.

Tel: 49-(228) 376751 Fax: 49-(228) 378178

http: //www. gi-ev. de

43. Greek Computer Society

Add: 2 Mavromichali Street, 106 79 ATHENS. Greece.

Tel: 30-(1) 3645274 Fax: 30-(1) 3645154

http: //www. otenet. gr /epy /index-en. htm

44. ICIMAF

Add: . c /o Mr B. J. Ferro, Academia de Ciencias de Cuba, Calle 15 No 551, Vedado, C. Habana
10400. Rep. de Cuba.

E-mail: icimaf @ ceniai. cu

Telex: 512230 icimaf cu

45. IEEE Computer Society (IEEE CS)

成立于 1951 年.

Add: 1730 Massachusetts Ave. NW, Washington, DC20036-1903. U. S. A.

Tel: 1-(202)371-0101 Fax: 1-(202) 728-9614

E-mail: hg. ofc @ Compmail. com

http: //www. computer. org

46. Indonesian Computer Society

成立于 1974 年.

Add: P. O. Box 2454, Jakarta 10002, Indonesian.

Tel: 62-(21) 5201010 Fax: 62-(21) 5200077

Telex: 62779.

<http://indonesian-society.com>

47. Information Processing Association of Israel (IPA)

Add: Kfar-Maccabiah, RAMAT-GAN 52109. Israel.

Tel: 97-(23) 715770 or 72 Fax: 97-(23) 5744374

48. Information Processing Society of Japan (IPSJ), 情报处理学会

Add: 108 东京都港区芝浦 3-16-20, 芝浦前川 Building, 7 阶. 日本国.

Tel: 81-(03)5484-3535 Fax: 81-(03)5484-3534

E-mail: edit j @ ipsj. or. jp

<http://www.ipsj.or.jp>

49. Institute of Electrical Engineers (IEE)

Add: Savoy Placa, London WC2R OBL UK. United Kingdom.

Tel: 44-0171-240 1871 Fax: 44-0171-240 7735

Telex: 261176 IEEDN G.

<http://www.iee.org>

50. Institute of Electrical and Electronic Engineers (IEEE)

Add: 345 East 47th, New York NY 10017-2394 U. S. A.

<http://www.ieee.org>

51. International Association for Mathematics and Computer in Simulation (IMACS)

Add: c/o Dept. of Computer Science, Rutgers Univ. New Brunswick, NJ 08903. U. S. A.

Tel: 1-(201)932 3998

E-mail: vichneve @ cs. rutgers. edu

<http://www.cs.rutgers.edu>

52. International Association for Pattern Recognition (IAPR)

Add: 66 Weston Park, Thames Ditton, Surrey KT7 OHL UK. United Kingdom.

Tel /Fax: 44-181 (398) 2766

E-mail: 100042.511 @ compuserve.com

<http://peipa.essex.ac.uk/iapr>

53. International Association for Statist. Computing

Add: c/s ISI, 428 Prinses Beatrixlaan, NL-2270 AZ VOORBURG. Netherlands.

Tel: 31-(70) 33757737 Fax: 31-(70) 3860025

E-mail: lieves @ cs. vu. nl

<http://fisher.stat.unipg.it/isac>

54. International Federation for Information Processing (IFIP)

成立于1960年1月,它是多国相关学会的联合学术团体,到1992年1月已有42个国家和地区的学术团体成为会员,每个国家只吸收一个会员单位。

Add: Hofstrasse 3, A-2361 Laxenburg, Austria.

Tel: 43-2236-73616 Fax: 43-2236-73616

E-mail: ifip @ ifip. or. at

http: //www. ifip. or. at

55. International Federation of Automatic Control (IFAC)

Add: Schlossplatz 12, A-2361 Laxenburg, Austria.

Fax: 43-2236-72859

E-mail: ifaca @ serv. univie ac. at

http: //www. ifac-control. org

56. International Medical Informatics Association (IMIA)

Add: 16 Place longemalle, CH-1204 GENEVA. Switzerland.

Tel: 41-(33) 282649 Fax: 41-(22) 7812322

E-mail: IFIP @ CGEUGE51. bitnet

http: //www. imia. org

57. International Society for Optical Engineering

Add: P. O. Box 10, Bellingham, Washington 98227-0010, U. S. A.

Tel: 1-(360) 676-3290 Fax: 1-(360) 647-1445

http: //www. spie. org

58. Irish Computer Society

Add: Dundrum Castle, Ballinter Rd. DUBLIN 16. Ireland.

Tel: 353-(1) 982692 Fax: 353-(1) 290016

59. Island Society for Information Processing

Add: Box 681, 121 REYKJAVIK. Iceland.

Tel: 354-(1) 27577 Fax: 354-(1) 25380

60. John v. Neuman Society for Computing Sciences

Add: P. O. Box 240, Bathori u. 16, H-1368 BUDAPEST. Hungary.

Tel: 36-(1) 1329349 or 1329390 Fax: 36-(1) 1318140

E-mail: huug @ neumann. h

61. Korea Information Science society (KISS)

Add: Room 401, Murijae Bldg. 984-1, Bangbae-3 dong, Seocho-ku, SEOUL 137 063, Rep. of Korea.

Tel: 82-(2) 5889246 Fax: 82-(2) 5889247

Telex: 22974 qnixsel.

62. Malaysian National Computer Confederation

Add: 46a, Jalan SS2 /66, Selangor Darul Ehsan, 47300 PETALING JAYA. Malaysia.

Tel: 60-(3) 7765160 Fax: 60-(3) 7747026

63. Nederlands Genootschap voor Informatica

Add: Van Diemenstraat 184, NL-1013 CP AMSTERDAM. Netherlands.

Tel: 31-(20) 6203676 Fax: 31-(20) 6203669

64. New Zealand Computer Society

Add: P. O. Box 12-249, WELLINGTON. New Zealand.

Tel: 64-(4) 731043 Fax: 64-(4) 731043

65. Norwegian Computer Society

Add: P. O. Box 6714, Rodelokka, N-0503 OSLO 5. Norway.

Tel: 47-(2) 370213 Fax: 47-(2) 354669

66. Pattern Recognition Society

Add: 3900 Reservoir Road, N. W. Washington, DC 20007. U. S. A.

67. Philippine Computer Society

成立于1967年.

Add: Penthouse c. Rivilla Building, Aguirre Street, Legaspi Viliage, Makati, Metro Manila, Philippines.

Tel: 63-(2) 8180381, 8184227

68. SADIO

Add: URUGUAY 252- 2“D”, 1015 BUENOS AIRES. Argentina.

Tel: 54-(1) 405755 or 453950 Fax: 54-(1) 111877

E-mail: jaiio @ sadio. edu

http: //www. uba. ar /wwws /sadio

69. Singapore Computer Society (SCS)

成立于1957年.

Add: 0511, 71 Science Park Drive, NCB Building. Singapore.

Tel: 65-7783901 Fax: 65-7788221

Telex: NCB RS 38610.

70. Society for Computer Simulation (SCS)

Add: P. O. Box17900, San Diego, CA92177-7900. U. S. A.

Tel: 1-(619) 277-3888

http: //www. scs. org

71. Society for Industrial and Applied Mathematics (SIAM)

成立于1952年.

Add: 3600 Univ. City, Science Center, Philadelphia, PA 19104-2688, U. S. A.

http: //www. siam. org

72. Society for Manegement Information Systems (SMIS)

成立于1968年.

Add: 111 East, Wacker Drive, Suite 600, Chicago, ILL 60601. U. S. A.

http: //www. cba. uga. edu /smis

73. South East Asia Regional Computer Confederation (SEARCC)

Add: c /o National Computer Board, NCB Building, 71 Science Park Drive, SINGAPORE 0511, Singapore.

Tel: 65-7783901 Fax: 65-7788221

Telex: 38610.

http: //www. searcc. org

74. SUCESU-NACIONAL

Add: Av. W-3 Norte, Quadra 504, Ed. Mariana, Sala 205, 70730 BRASILIA, Brazil.

Tel: 55-(41) 2256373 (Secr.) Fax: 55-(41) 2282177

Telex: 272161.

http: //www. internetional. com. br /sucesu

75. Swedish International Federation for Information Processing (SIFIP)

Add: Box 22114, S-104 22 STOCKOLM. Sweden.

Tel: 46-(8) 6520720 Fax: 46-(8) 6500343

76. Swiss Federation for Information Processing Societies (SVI /FSI)

Add: P. O. Box 71, CH-8037 ZURICH. Switzerland.

Tel: 41-(01) 2757121 Fax: 41-(01) 2727912

77. West African Regional Computer Society (WARCS)

Add: c /o Dr. O. C. Akinyokun, P. O. Box 7600, LAGOS, Nigeria.

Tel: 234-(36) 230747 or 230312

Telex: 34265 cepeda ng.

78. Word Computer Graphics Association (WCGA)

Add: 2033 M St. , Suite 399, Washington, DC 20036. U. S. A.

Tel: 1-(202) 715-9556